Aalto University
School of Engineering

Using Social Media Photo Metadata for Finding Interesting Places

Master's Thesis
Department of Real Estate, Planning and Geoinformatics, School of Engineering, Aalto University

Espoo, 4 May 2015

Joona Heinikoski
Master of Science in Technology

Supervisor: Professor Kirsi Virrantaus
Instructor: D.Sc. (Tech) Jussi Nikander

Aalto-yliopisto
Insinööritieteiden korkeakoulu

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Joona Heinikoski		
Työn nimi Sosiaalisen median valokuvien metatiedon käyttö kiinnostavien paikkojen etsinnässä		
Koulutusohjelma Geomatiikka		
Pääaine Geoinformatiikka		**Koodi** M3002
Työn valvoja Professori Kirsi Virrantaus		
Työn ohjaaja Tekniikan tohtori Jussi Nikander		
Päivämäärä 4.5.2015	**Sivumäärä** 39 + 10	**Kieli** Englanti

Tiivistelmä

Tämä työ tutkii kuinka sosiaalisen median valokuvien paikkatietoa hyödyntämällä voidaan etsiä mielenkiintoisia paikkoja. Tarkemmin tavoitteena on löytää erityisesti turisteille mielenkiintoisia vierailukohteita. Koska nykyaikaisissa mobiililaitteissa on laadukas kamera ja nopeat datayhteydet, erilaisissa kuvapalveluissa on valtavat määrät käyttäjien ottamia valokuvia. Tässä työssä hyödynnetään ja vertaillaan klusterointimenetelmiä tiedon louhinnaksi tästä valtavasta lähtöaineistosta. Työssä käytetyt klusterointimentelmät ovat K-Means ja DJ-Cluster. Pelkästään sijainniltaan mielenkiintoisten paikkojen lisäksi datasta etsitään paikkoja, jotka ovat mielenkiintoisia johonkin tiettyyn aikaan vuodesta.

Työn teoreettisessa osassa esitellään tarpeelliset menetelmät kiinnostavien paikkojen löytämiseksi. Kaikki menetelmät implementoitiin Python-ympäristössä osana työtä ja esitellään tarvittavalla tarkkuudella, jotta lukija pystyisi implementoimaan menetelmät. Työn kannalta tärkeimmät käytetyt menetelmät ovat klusterointimenetelmät, etäisyysperusteinen K-Means ja tiheysperusteinen DJ-Cluster. Etäisyysperusteiset klusterointimenetelmät vaativat kaikkien eri data-akseleiden etäisyyksien olevan verrattavissa toisiinsa. Tätä varten käytetään z-score normalisointimenetelmää, jolla erilaiset numeeriset muuttujat voidaan muuttaa toisiinsa verrattavalle asteikolle. Klusterointia varten kaikkien arvojen etäisyyksiä ja keskiarvoja tulee pystyä laskemaan. Perinteiset lineaariset menetelmät eivät sovellu syklisille arvoille, joten aikaa käsitellään suunnan kaltaisesti yksikkövektoreita soveltaen.

Työn lopputulokset ovat tarkoitettu myös alaan perehtymättömän käyttäjän tutkittavaksi. Jotta tulosten selaaminen olisi mahdollisimman esteetöntä, työn visualisointijärjestelmä rakennettiin toimimaan internetselaimessa. Visualisoinnissa pyritään esittämään tärkeimmät klustereiden spatiotemporaaliset arvot yhdellä vilkaisulla. Tätä varten kartalla esitetään tutkakaavion kaltaisella symbolilla kuinka paljon pisteitä klusterissa on kuukausittain.

Työn menetelmiä sovelletaan Flickr-palvelusta hankittuun 121 323 pisteen esimerkkiaineistoon. Aineisto sijoittuu Japaniin Osakan alueelle. Klusteroinnin tuloksia arvioidaan sekä laskennallisin mittarein, että laadullisin menetelmin. Tuloksista käy ilmi DJ-Cluster menetelmän olevan kiistattomasti tavoitteeseen paremmin soveltuva menetelmä.

Avainsanat GIS, klusterointi, sosiaalinen media, syklinen aika, tiedonlouhinta, visualisointi

Aalto University
School of Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi

Abstract of master's thesis

Author	Joona Heinikoski
Title of thesis	Using Social Media Photo Metadata for Finding Interesting Places
Degree programme	Degree Programme in Geomatics
Major	Geoinformatics **Code** M3002
Thesis supervisor	Professor Kirsi Virrantaus
Thesis advisor	D.Sc. (Tech) Jussi Nikander
Date	4.5.2015 **Number of pages** 39 + 10 **Language** English

Abstract

This thesis explores how geotagged social media photographs can be utilised for finding interesting places. More specifically, the goal is to find interesting places for tourists to visit. As modern mobile devices are equipped with high quality cameras and fast data connections, different photo sharing services have large collections of photos posted by users. This work employs and compares different clustering methods for mining data out of this large dataset. The two adopted clustering methods are K-Means and DJ-Cluster. Along with only spatially interesting places, the data was searched for places that are only seasonally interesting.

The theoretical section of this thesis presents the necessary methods for finding these interesting places. All methods were implemented in Python environment as part of the research work and are presented in such detail to enable implementation by the reader. The most important methods in this work are the two clustering methods, distance-based K-Means and density-based DJ-Cluster. Distance-based methods require that distances in all data-axes are comparable with each other. To enable this, a z-score normalization method is utilised. Z-score normalizes different numerical variables to an equal scale. Clustering also requires calculating the distances and averages for different variables. Traditional linear methods cannot be used with cyclical time. Thus, cyclical time is processed similarly to direction using unit vectors.

The results of the research are intended to be also understandable for non-specialist users. To enable exploring of the results as easy as possible, a visualization system was built to for use in any internet browser. The data visualization aims to present every cluster's most important spatiotemporal variables at a glance. To achieve this, the map symbol for clusters is a modified radar plot, which presents the amount of points in different months.

The methods are applied to the 121 323 point sample dataset acquired from Flickr service. The dataset is located around Osaka, Japan. The clustering results are evaluated with both statistical and qualitative methods. The results indicate that DJ-Cluster is a far superior method for the goal of finding interesting places both spatially and temporally.

Keywords clustering, cyclical time, data mining, GIS, social media, visualization

Acknowledgement

First, I would like express my gratitude to my supervisor, Professor Kirsi Virrantaus, for all the inspiring education that the Department of Real Estate, Planning and Geoinformatics has provided for me during my studies.

Next, I want to thank all the other students and teachers I've had the pleasure to work with along the course of my studies. I have been able to meet people with different backgrounds and make many new friends. You have truly given me the motivation to pursue my studies to conclusion. I would like to mention by name Jaakko Rantala and Kamyar Hasanzadeh who helped me with some of the methods used in this work.

Even though I have completed this work completely on my own time without any outside funding, I would also like thank my current employer, Housemarque Oy, and their CEO, Ilari Kuittinen, for allowing me to continue and complete my studies along my job in the company. Everyone in the company has always been very supportive of my studies. I would like to especially thank Juha Riihimäki and Markku Velinen for your interest in my thesis project and for listening to my ranting about the bugs in some random open-source libraries I was trying to integrate in my codebase or the quirks in different APIs I was using at the time.

I would also like to thank my girlfriend Charlotta Tiuri and my family for always encouraging me during my studies and this thesis project. With your support I have been able to complete everything in reasonable time.

Finally, I would like to thank maybe the most important supporter of this thesis, my instructor, Jussi Nikander. Even if you do not work in the Aalto University any more, you always answered my questions in timely manner and provided invaluable comments for this thesis. Without your guidance, I might not have been able to complete this thesis.

Espoo, 4.5.2015
Joona Heinikoski

Table of Contents

Abbreviations

API	Application Programming Interface
AVG	Average
CH	Calinski-Harabasz [Criterion]
GIS	Geographic Information System/Science
MAD	Mean Absolute Deviation

List of Figures

List of Tables

1 Introduction

1.1 Background

Data mining methods have potential to reveal new knowledge from large and unorganized datasets. Large collections of social media data is a good example of such dataset. The recent rise of perpetually internet connected mobile devices equipped with high quality cameras has resulted in people taking and publishing more photos in social media than ever before. The time it takes to publish photos online after taking them can be only as little as few seconds. Ever growing collections of photos of virtually any subject are therefore readily available for anyone to browse. By the end of 2014, over 60 million photos were published every month just in photo sharing service Flickr alone (Michel, 2015).

A popular occasion for taking photos and posting them is while travelling. Different photo services are filled with tourist photos. Logically, this kind of massive dataset could be used to find interesting places for other people planning a trip to a specific area. Most modern mobile devices are also equipped with a real-time location service, usually a GPS receiver. Whenever taking a photo, the device can automatically save the coordinates where it was taken to the photo's EXIF-metadata. While posting photos for their friends and anyone in the internet to see, ordinary people are actually producing huge amounts of relatively accurate geographic data as by-product. By tapping into this information, people's holiday photos could become viable source of data for GIS analysis.

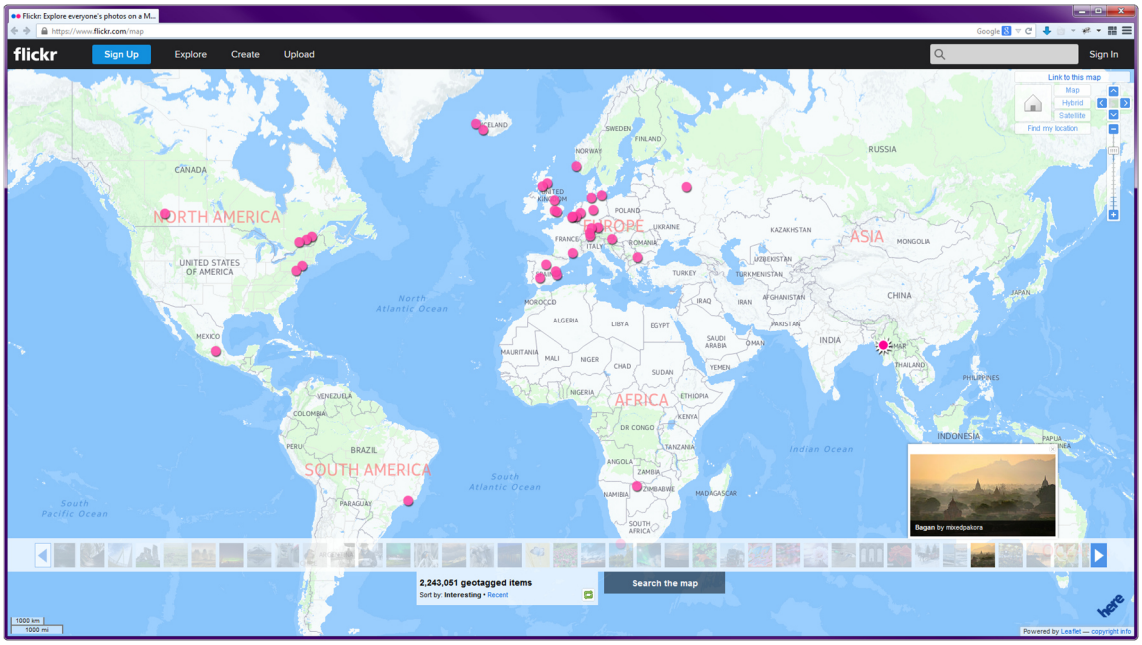


Figure 1. Flickr offers a basic map view for browsing interesting geotagged photos (accessed April 14 2015)

To enable fast adoption of their social media services, many service operators offer free and open APIs (Application Programming Interface) for third party applications to ac-

cess and post to the services. Flickr, for example, offers very easy to use http-based REST API for posting new photos and querying existing pictures with many different criteria. Photos can be queried for example by user-added tags or by the date when they were taken. Photos with location metadata can also be queried within a specified bounding box (Flickr API, 2014). Different API's make it convenient for any computer application to gain access to various social media entries. Along with less serious applications, such as ones used for posting personal status updates, APIs can also be used for acquiring source data for scientific analysis.

1.2 Objective and Research Questions

The main objective of this research is to study how information of interesting places can be extracted from metadata of photos posted to social media. In this context interesting places are locations that many people find appealing enough to photograph and post to social media. The intended audience who would be interested in these places are people planning a trip to new destination. The anticipated results of this research should preferably point out many almost point-like singular places with significant amount of photos taken by different people instead of large areas containing potentially many interesting places. Another objective is to find places that are appealing during different times of the year.

The research questions for this thesis are

- What methods can be used for finding interesting places in social media photograph metadata?

Before analysing the data, this thesis presents theory and methods used for extracting the interesting places in large datasets.

- What is the most suitable method for processing the data?

This thesis presents two different methods and two variants of these methods for finding interesting places. The methods are analysed and compared to find out how suitable they are for the research goal.

- What kind of interesting places can be found in dataset of geo-tagged social media photographs?

Analysis results for a sample dataset are reviewed and evaluated as part of the work. One of the features evaluated is what kinds of places the analysis has found. Some particularly interesting places are highlighted.

- Can seasonally interesting places be found in the data?

Prior research on similar subjects has mainly considered the location of the taken photographs. This thesis researches if time of year can be included in search for interesting places and if there is seasonal variation in the results.

1.3 Dataset

The research area was chosen to be a roughly 14 000 km^2 area around Osaka, Japan. Area was chosen due to high availability of photos and many popular tourist spots, such as castles and shopping areas. The research dataset was acquired from Flickr through their http REST API interface. The chosen photos were limited to be taken earliest in 2010. For each month, maximum of first 2500 photos with highest "interestingness" value were chosen. Flickr calculates an "interestingness" value for photos with an undisclosed algorithm. According to their blog posts, interestingness is based on the popularity of both photographer and the specific photo, comments posted on the photo and tags associated with it, among other elements (Flickr, 2014). While not comprehensive measure, ordering photos with interestingness should increase the variation of different kinds of photos from many users for the photos that are used in the research.

The people who had taken the photos were not informed about using their photos in research. All of the used photos and their metadata had been marked as public by the uploader, so there should not be any ethical problems using them for research purposes. Additionally, the usernames were saved in hashed form in order to further obfuscate them. Some sort of user ID was needed in the analysis for mitigating the effect of mass uploaders.

In addition to the location where the photo had been taken, time and date, user-entered tags and md5 hash of the username were saved in the database. This resulted in 121 323 photo entries, 58 538 unique tags and 938 165 tags associated with photos. As no actual image data was saved, the database size for the sample data was about 122 MiB.

1.4 Analysis Technical Implementation

The analysis result was designed to be useful for non-specialist audience, so exploring it should be as approachable as possible. Hence, an internet browser based solution was chosen for data presentation. The analysis methods were custom-implemented in Python language using different open source tools and libraries.

To enable easy transition from analysis to browser-based visualization, both the analysis system and presentation service were built on Python-based Django[1] web server platform. Being a full-fledged web server, Django provides template-based dynamic page rendering and http-request routing, along other useful features for serving and displaying the results.

For research implementation Django provides easy-to-use database abstraction layer. Employing GeoDjango[2] spatial data extension, SciPy[3] numeric computing library, GE-

[1] https://www.djangoproject.com/

[2] https://docs.djangoproject.com/en/1.8/ref/contrib/gis/

[3] http://www.scipy.org/

OS[1] geometry library and GDAL[2] geospatial data abstraction library, we have sophisticated enough tools for building the analysis algorithms.

Research data was stored in a MySQL[3] database running MyISAM database engine, which has basic support for geographic data. MySQL database was chosen for being readily available and well supported within Django platform. MySQL's geographic functionality is somewhat limited, but its R-tree indexed bounding box query provided fast enough access to the data.

All of the software created for the analysis is freely available as MIT-licensed open source in project's git repository[4].

1.5 Thesis Structure

Following the Introduction chapter, the next chapter in this thesis is Theoretical Background. This chapter is intended to equip the reader with adequate understanding of concepts and methods used to achieve the results. The chapter explores for example different clustering methods and how to process time as a cyclical value.

The third chapter, Data Analysis, presents the conducted analysis and its results. The research dataset and four different analysis methods are presented in detail. Before exploring the results, reader is made familiar with the internet browser based visualization system and used multidimensional visualization methods. After all these prerequisites are established, the actual results are reviewed and presented in detail. First, statistical metrics for different methods are evaluated and compared between each other. The statistical metrics already suggest what kind results different methods have produced and next the results for each method are presented one by one. This includes overview maps, zoomed in detail maps and qualitative evaluation of the results. In spatiotemporal DJ-Cluster results, special attention is given to clusters with particularly interesting seasonal effect.

Further, in Discussion chapter, the results are compared and used methods are evaluated. This chapter discusses how feasible a publicly available system built on basis of this research could be implemented and what additional research should be conducted. Finally, Conclusions chapter summarizes everything presented in this thesis.

---

[1] http://trac.osgeo.org/geos/

[2] http://www.gdal.org/

[3] http://www.mysql.com/

[4] https://github.com/joonamo/photoplaces/

2 Theoretical Background

This chapter presents the theoretical background needed to carry out the work. The used dataset doesn't inherently include the features this work aims to find. Therefore, the desired results couldn't be found with traditional statistic methods and thus data mining methods are employed. The main data mining method is clustering. Two different clustering paradigms, distance-based and density-based clustering, are introduced and one method from both is presented in detail. These methods are distance-based K-Means clustering (MacQueen, 1967) with K-Means++ initialization (Arthur and Vassilvitskii 2007) and density-based DJ-Cluster (Zhou et al. 2007).

As time-axis is considered in distance-based clustering, the multidimensional data has to be normalized. In this work, the used normalization method was z-score. Z-score is introduced in chapter 2.2. In this research, time was treated as a cyclical variable. Later in this chapter some methods for analysing cyclical variables are explored.

As the data source is open and public service, any user can upload as many or few photos they want. This might skew the results unfairly. Methods to prevent few enthusiastic users from affecting the data significantly are discussed. Finally, a numerical method for evaluating clustering fitness is presented.

None of the methods used were available in the used numerical computing libraries and were implemented as part of the research project.

2.1 Clustering Methods

The primary analysis of the data was conducted with clustering methods. Berkhin (2002) describes clustering as "-- a division of data into groups of similar objects. Representing the data by fewer clusters necessarily loses certain fine details, but achieves simplification. It [clustering] models data by its clusters." In other words, clustering simplifies dataset to groups of data. Clustering can be carried out for any numeric data and even categorical data, but is especially intuitive for geographic data.

In this research, the time-axis was also included in the clustering analysis. Some of the used clustering methods require measuring and comparing distances over all axes. Time and space being inherently unsuitable to be compared requires data to be normalized. After normalizing variables, distance of two data points in normalized time can be compared to distance in normalized spatial space. Normalization was achieved using z-score value, which describes how far from mean the data point is in normalized unit. Two different clustering methods, K-Means and DJ-Cluster, were used and compared in the analysis. Analysis was also conducted with and without time variable to see if seasonal changes affect the results.

2.1.1 Distance-Based Clustering, K-Means Clustering

K-Means is a widely used and extensively researched distance-based clustering method. It was first introduced by MacQueen (1967). As the name implies, distance-based clustering methods assign points to clusters based on the distance between point and cluster

centre. In practice, K-Means seeks to minimize sum of squares between cluster points and its centre. Any distance measure compatible with this can be used. In this project simple Euclidian distance was used.

A distinct feature of K-Means clustering is that it will always create exactly as many clusters as defined in the sole input parameter, the *k*-value. K-Means algorithm works iteratively by assigning data points to different clusters and updating the cluster centre points (Maimon and Rokach, 2010). Initially, *k* cluster centres are chosen according to some initialization method. Many different initialization methods exist. The initialization greatly affects the end results of the algorithm and an unfortunate choice of initialization method may produce worthless results. Traditional initialization methods include Forgy and Random Partition methods (Hamerly and Elkan, 2002). In Forgy method, the location of *k* points randomly chosen among the data is set as the initial cluster centres. Random Partition first randomly assigns every data point into one of the *k* clusters and then proceeds to calculate the means for centres. It is characteristic for Random Partition to initially clump the cluster centres close the global mean of the data.

For this analysis, initialization method presented by Arthur and Vassilvitskii (2007) called K-Means++ was chosen. K-Means++ initialization has similarities with Forgy method, but it works iteratively and uses weighted random selection. The first cluster centre is chosen randomly among the data points. Additional cluster centres are chosen by weighted random distribution. Probability of selecting the location of a data point as cluster centre is proportional to squared distance to closest already chosen cluster centre, formally

$$p(x) = \frac{D(x)^2}{\sum_{x \in X} D(x)^2}, \tag{1}$$

where D(x) is the distance of point to the closest already selected cluster centre. According to empirical results, K-Means++ both improves quality of clustering results and reduces time for algorithm to converge compared to Random Assignment or Forgy initialization.

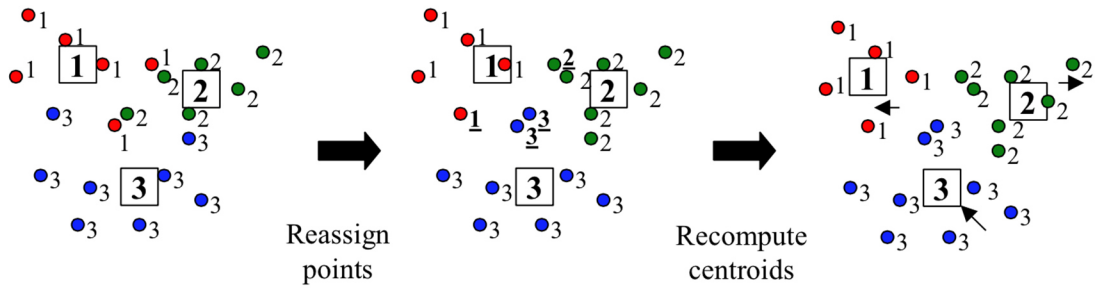


Figure 2. K-Means algorithm works iteratively in two phases (Berkhin, 2002)

After initialization, K-Means clustering works iteratively in two phases; reassign points and recomputed centres (Figure 2). In reassign points phase, the closest cluster centre is searched for each data point and the point is assigned to belong in that cluster. In the recomputed centres phase, cluster centres are updated to be in the mean of their points. By iterating these two steps, the centres crawl towards the local optimum. Iterating

these two steps is continued until ending condition is met. K-Means is computationally NP-Hard (Mahajan, Nimbhorkar and Varadarajan, 2012) and many different ending conditions have been proposed. Different ending conditions have varying reduced requirements and accuracies. Traditional and very strict ending condition requires that none of the data point changes the cluster they belong to in reassign points phase. More relaxed conditions for example set the maximum count of points changing their cluster or maximum distance cluster centre may move (Virrantaus, 2013). K-Means clustering is presented in pseudo code in the Figure 3.

```
// Initialization (K-Means++)
centres = []
centres.append(random_choice(sample))
while centres.size < k:
    weights = []
    for point in sample:
        weights.push(dist_to_closest(point, centres) ** 2)
    centres.push(random_choice_weighted(sample, weights))

// Iteration (Regular K-Means)
while not ending_condition_met():
    for point in sample:
        closest_centre = find_closest(point, centres)
        set_cluster(point, closest_centre)
    for centre in centres:
        centre.position = avg(centre.points)
```

Figure 3. K-Means algorithm with K-Means++ initialization

K-Means is widely used as it is thoroughly analysed and simple to understand and implement (Berkhin, 2002). Still, K-Means clustering has been criticized for example for being unsuitable for highly uniform data sets and sometimes finding skewed clusters (Zhou et al. 2007), (Crandall et al., 2009). On the other hand, depending on used dataset, research has found K-Means to produce better results than density-based clustering algorithms (Nikander, Kantola and Virrantaus, 2011).

2.1.2 Density-Based Clustering, DJ-Cluster

Another popular clustering approach is density-based clustering. In spatial and spatio-temporal space, density-based clustering methods examine density-based connectivity of data points. The connectivity is defined in the density neighbourhood function. One of the most well-known density-based clustering method is DBSCAN (Density Based Spatial Clustering of Applications with Noise). DBSCAN works by analysing density neighbourhoods for all points in the dataset. The density neighbourhood of DBSCAN is defined with two input parameters, *eps* and *min_pts*. Two points have density-based connectivity if distance between them is less than *eps*. A component of cluster is found if there are more than *min_pts* points within *eps* range of a point. Clustering then recursively analyses density neighbourhoods of all points in the neighbourhood and grows to include acceptable neighbourhoods in the cluster. Points failing to have acceptable neighbourhood are marked as noise. (Berkhin, 2002)

One the biggest difference between most density-based and most distance-based methods is that density-based methods can find clusters of arbitrary shape (Figure 4) and may or may not include all data points in the resulting clusters. Points classified as noise are not included in the final clusters, which may affect the results drastically. Especially ability to discard noise is very beneficial for finding concentrated interesting places (Kisilevich et al., 2010). Also, the amount of resulting clusters from density-based methods is not known before run, unlike K-Means and its *k*-value that defines the amount of output clusters. Another difference between density-based methods and many distance-based methods is that density-based methods usually require only one pass over the dataset instead of multiple iterations. This might result in faster computational time for completing clustering especially compared to NP-Hard K-Means. The performance of density-based methods is often mostly influenced by how well the data is indexed and how fast the points in range can be queried. (Berkhin, 2002)

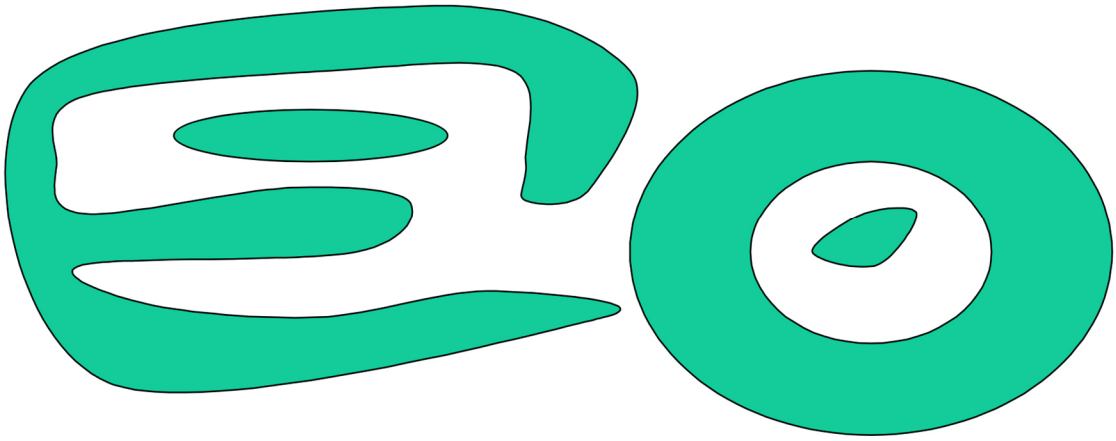
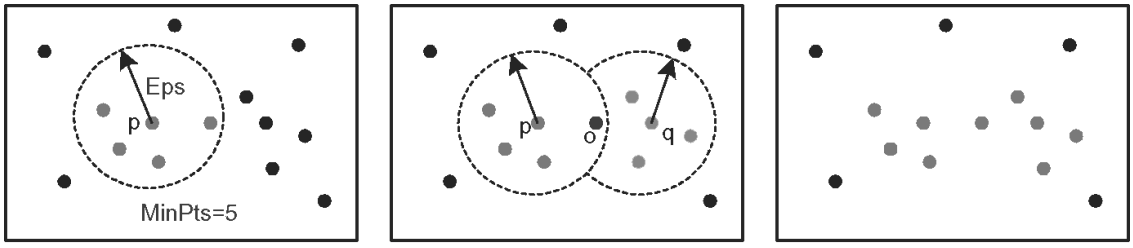


Figure 4. As clusters in density-based clustering grow to include points in neighbourhood, clusters of arbitrary shape can be found (Berkhin, 2002)

DJ-Cluster is another density-based clustering method presented by Zhou et al. (2007). DJ-Cluster is based on same principles as DBSCAN, but improves on more efficient memory-usage and potentially faster running time. The name DJ-Cluster comes from Density and Join concepts it is relies on.



(a) $Density - N(p)$ (b) $Density - Joinable$ (c) $DJ - Cluster$

Figure 5. DJ-Cluster Concepts (Zhou et al. 2007) a) Density neighbourhood is similar to DBSCAN density neighbourhood b) Clusters are joined together if a point in neighbourhood already belongs to a cluster c) Joined cluster

DJ-Cluster algorithm works by calculating the density neighbourhood for every point in the dataset. The density neighbourhood of a point is defined similarly as in DBSCAN. Neighbourhood of a point consists of all other points within range of *eps*. Neighbour-

hood also has to contain equal or greater amount of points than *min_pts* value to be acceptable. Failing either of these requirements makes point's neighbourhood to be an empty set and point is classified as noise. If a point has valid neighbourhood, a new cluster is created with all points in the neighbourhood. Even points earlier classified as noise are included, if they are found in another point's neighbourhood. If one or more points in the neighbourhood already belong to another cluster, all points in clusters and neighbourhood are joined together creating a bigger cluster. DJ-Cluster concepts and joining of two clusters are presented in Figure 5. The algorithm is presented as pseudo code in Figure 6.

```
unprocessed = sample
while unprocessed is not empty:
    current = unprocessed.pop_front()
    neighborhood = DJNeighbourhood(current, eps, numpts)
    if neighborhood is empty:
        mark current as noise
    elif neighborhood contains cluster:
        join neighborhood and each cluster in neighborhood
    else:
        create cluster from neighborhood
```

Figure 6. DJ-Cluster algorithm

2.2 Normalizing Data with Z-Score

Spatial and temporal data are by nature very different and not comparable with each other. Just by comparing two values, it is for example impossible to say whether distance of 100 meters is greater or less than a difference of two months. Distance-based clustering works by measuring the distance of data points to cluster centres over all axes, so all of the values need to be comparable with each other. Both time and location variables are continuous data, so they can be normalized to standardized scale and be compared.

For this project, z-score normalization was employed for standardizing the data. The z-score describes the point's normalized distance from mean of that axis. Z-score also takes into account the order of values. Therefore values greater than mean will be positive and values less than mean negative. For calculating z-score, a deviation variable and the mean of data need to be calculated. The deviation variable and mean are calculated individually for all axes. (Kaufman and Rousseeuw, 2009)

Any deviation variable can be used for calculating z-score, but Kaufman and Rousseeuw (2009) recommend mean absolute deviation (MAD). MAD measures the mean of distance between all points and the mean of the axis. MAD is conceptually easy to understand and isn't affected by small amount of even large outlier values unlike standard deviation. Mean absolute deviation is defined as

$$s = \frac{1}{n}\left(\sum_{i=1}^{n} |x_i - m|\right), \tag{2}$$

where x_i is the i^{th} value on the axis and m is mean of axis.

For each axes of data point, the z-score is calculated as

$$z_i = \frac{x_i - m}{s}, \tag{3}$$

where s is the deviation of the axis and m is the mean of the axis.

2.3 Time as Cyclical Variable

Time values are often considered as 1-dimensional continuous value with the resolution of for example day, month or SI-standard second. In this case, the distance between two values is calculated similarly as any other continuous value,

$$d_{ij} = |x_i - x_j|. \tag{4}$$

In this analysis, we are interested to find if there is seasonal change in the data. Thus time has to be considered as a cyclical value. This complicates calculation involving time variables a bit. By using the regular 1-dimensional distance, the distance between December and January would result in |12 - 1| = 11 months, when intuitively they should be just one month apart. A different approach has to be taken when calculating distances of cyclical time. Distance in cyclical values, including time, can be calculated with

$$d_{ij} = \min\{|x_i - x_j|, c - |x_i - x_j|\}, \tag{5}$$

where c is the length of the cycle. (Tango, 1984)

2.3.1 Cyclical Mean

Similarly to distance of cyclical time values, mean of cyclical values cannot be calculated using traditional statistical functions. A common example of cyclical variables are angles. Degrees are a commonly used arbitrary measure of angles. Degrees divide unit circle to 360 equal partitions for describing different angles. They can be converted to standard mathematical rotation, also known as radian, with equation

$$\theta = \alpha * \frac{2\pi}{360°}, \tag{6}$$

where θ is angle in radians and α is the angle in degrees (Seppänen et al., 2005). Similarly any arbitrary cyclical value can be converted to radians by

$$\theta = v * \frac{2\pi}{c}, \tag{7}$$

where v is the cyclical value and c is the cycle length. For example, by applying this equation with cycle length 12, months can be converted to radians. Further, any radian rotation can be converted to unit vector $\dot{q}$ by applying equation

$$\dot{q} = (cos\theta, sin\theta). \tag{8}$$

The unit vectors can then be summed together and normalized back to unit vector to approximate the average direction in unit vector form. If required, weights can easily be added to the equation by multiplying the unit vector with the corresponding weight. Weighted mean vector is defined as

$$\dot{q}_m = \sum_i^n w_i \dot{q}_i \Big/ \left| \sum_i^n w_i \dot{q}_i \right|, \tag{9}$$

where w_i is the weight for the i^{th} value (Gramkow, 2001). The resulting unit vector can be converted back to radian angle using common numerical computing function arctan2 and from radian back to cyclical values, such as months, by using equation

$$v = \theta * \frac{c}{2\pi}. \tag{10}$$

2.3.2 Normalizing Time Variable

As discussed earlier, mean absolute deviation (MAD) measures average distance from centre of the set. Standard MAD equation doesn't work with cyclical values, but with some changes it can be made compatible with them. MAD measures the distance between the mean of data and a data point. A generalized mean absolute deviation can be written as

$$s = \frac{1}{n}\left(\sum_{i=1}^{n} d(x_i, m) \right), \tag{11}$$

where d is distance function and m is the mean of data. Both of these are have already been defined for cyclical values.

Similarly, z-score can be generalized by using distance and the mean of the set. The distance function always returns values equal or greater than zero, but z-score takes into account the order of values. Hence the sign of values whose distance is calculated using $c - |x - m|$ has to be negated. The mean can be considered to shift the arbitrary starting and end points of the cycle. Z-score for cyclical values can be calculated by

$$z_i = \frac{k * d(x_i, m)}{s}, k = \begin{cases} sgn(x_i - m), & if\ |x_i - m| \leq\ c - |x_i - m| \\ -sgn(x_i - m), & if |x_i - m| >\ c - |x_i - m|' \end{cases} \tag{12}$$

where x_i is the value z-score is calculated for, m is the mean of the set and c is the length of the cycle. Function sgn is the signum function, which returns -1, 0 or 1 depending if the value is less, equal or greater than zero (Weisstein, 2014).

2.4 Mitigating the Effect of Mass Uploaders

The analysis data is based on public data source, where basically anyone can post as many or few data points as they will. This might skew some the clustering results towards some user's interest point from places that are popular among many people. Superficial review of the data revealed that one user, for example, had posted tens of pho-

tos of flowers in very close proximity and another had photographed rear view of a boat ride in few minute intervals. In worst-case scenario, even someone's home could be classified interesting place if enough photos were posted there.

For mitigating the effect of mass uploaders in the final analysis results, Crandall et al. (2009) recommend a small tweak to how clustering neighbourhoods are calculated. When counting how many points there are in the neighbourhood, the amount of different users is counted, instead of raw count of data points in the distance. This should shift the emphasis of the results to places that are interesting to many people, instead of just one enthusiastic photographer. This approach can only be used with clustering methods that process neighbourhoods, namely density-based clustering methods.

A derivative method was employed in distance-based clustering. When calculating cluster centres, every data point in cluster is given weight inversely proportional to how many photos from that user is currently included in the cluster. Formally the weights are defined as

$$w_i = \frac{1}{N_{User(i)}}, \tag{13}$$

where $N_{User(i)}$ is the count of photos in the current cluster from the user who produced data point i. In effect, this method gives every user total weight of one. This way high density of photos of a flowerbed from just one user has the same effect in the cluster as single photo of a popular tourist spot from single user.

2.5 Evaluating Clusters with Calinski-Harabasz Criterion

After the clustering has been performed, the results should be somehow evaluated. In this research, the main evaluation method was qualitative evaluation by the researcher, but a quantitative method was also employed.

Many different numerical evaluation methods for clustering results exist. One of them is Calinski-Harabasz Criterion that outputs a positive value for clustering. This value can be used to compare the fitness of different clustering runs and algorithms for a specific dataset. A greater value indicates better fitness, but cannot be used for comparing fitness of clustering of different datasets. The CH criterion is well suited for this research, as different clustering methods are performed for same dataset.

Calinski-Harabasz Criterion is defined as

$$CH = \frac{SS_B}{SS_W} \times \frac{(N-k)}{(k-1)}, \tag{14}$$

where SS_B is overall between-cluster variance, SS_W is overall within-cluster variance, N is the total number of input data points (including the ones possibly classified as noise), and k is the number of clusters.

SS_B, the between cluster variance, is defined as

$$SS_B = \sum_{i=1}^{k} n_i \|m_i - m\|^2, \tag{15}$$

where k is the number of clusters, n_i is the number of points in cluster i, m_i is the centroid of cluster i, m is the mean over all data and ‖…‖ denotes Euclidian distance.

SS_W, the overall within-cluster variance, is defined as

$$SS_W = \sum_{i=1}^{k} \sum_{x \in c_i} \|x - m_i\|^2, \tag{16}$$

where k is the number of clusters, x is a data point in cluster, c_i is cluster i, m_i is centroid of the cluster and ‖…‖ denotes Euclidian distance.

Calinski-Harabasz Criterion encourages many compact clusters. Higher value can be achieved by maximizing SS_B and minimizing SS_W. To achieve this, the points in one cluster should be very similar and clusters should be as different as possible from one-another (Maulik and Bandyopadhyay, 2002). The goal of the research is to find many small and distinct clusters, so CH Criterion is an appropriate choice for an evaluation method.

Regrettably, calculation of both SS_W and SS_B rely heavily on distance to mean, which is not as well defined for cyclical values as linear values. The resolution of time used in the analysis is also somewhat coarse, only the month is considered. Thus, clusters with values from months close to the poorly defined mean might get unfairly low SS_W. The data is also collected to include fairly equal amount of points from different months, further diminishing the meaning of temporal mean. Thus, CH Criterion was only calculated and evaluated for the spatial axis.

3 Data Analysis

This chapter describes the conducted analysis, the used dataset and the methods employed. The results produced by different methods are presented with maps. These results are analysed and commented on for both their qualitative and quantitative features. First, statistical properties of the results are analysed for all methods. Next, every method is qualitatively analysed by the maps they produced. Interesting clusters and occurrences among the results are highlighted.

3.1 Description of Analysis

As described earlier, the used dataset includes 121 323 photos around Kansai area in central Japan. The dataset covers large cities such as Osaka, Kobe, Himeji and parts of Southern Kyoto. This area was chosen as the sample area as it includes many popular tourist sights and hosts many seasonal events. It was assumed that a person planning a tourist trip could be surveying an area much like this. Summary of used input values and other notes for different methods are presented in Table 1.

Four different analyses were conducted for the data. Both K-Means++ initialized K-Means clustering and DJ-Cluster methods were run with and without taking temporal dimension into account. K-Means runs were run for normalized data, while DJ-Cluster was run for natural values. K-Means relies heavily on the distance measure and concept of mean centre, so all the different axes should be comparable with each other. Thus, values had to be normalized. Density-based clustering doesn't have this kind of requirement. The density neighbourhood function can be defined to handle different axes any way desired. Simple solution would be to query points within *eps* distance over all normalized axes. In the case of this work, temporal dimension was considerably lower resolution than spatial dimension. The used density neighbourhood function was defined to take two different *eps* values, *eps* for spatial axis and *eps_time* for temporal axis. This allowed more fine-tuned approach to handle the neighbourhood.

First analysis performed was spatial DJ-Cluster. It was initially run for a limited, but dense dataset located in central Osaka to find out suitable input variables. The goal in this project was to find small, but dense clusters. Thus, small *eps* value was used with relatively high *min_pts*. The used *eps* value of 0.001° and *min_pts* of 20 yielded good results in preliminary tests. These input variables were used to run the analysis for the complete data, which produced exactly 100 clusters.

Next, both of the K-Means analyses were performed. K-Means takes one input variable, the *k*-value, and always produces that many clusters. The earlier run of DJ-Cluster produced 100 promising clusters and this was used as basis for choosing the *k*-value. Standard K-Means clustering doesn't have concept of noise and will always use all data points in clusters. Using the exact cluster count as *k*-value would result in much larger clusters, so *k*-value was chosen to be 1.5 times the cluster count of clusters generated by DJ-Cluster. Both spatial-only and spatiotemporal runs were expected to result in somewhat similar clusters so similar *k*-value was used. For the K-Means++ initialization, only spatial coordinate was used for both runs.

Finally, spatiotemporal DJ-Cluster analysis was performed. The first DJ-Cluster run produced appropriately sized clusters, so the same *eps* value was used. For this run, a new variable, *eps_time,* was introduced. Unlike K-Means, spatiotemporal DJ-Cluster wasn't run with normalized data. Instead, the run was performed in natural space and absolute amount of month difference was used for defining temporal difference. The intent was to find clusters for locations interesting for only limited time of year. Normalized temporal axis had a lot more coarse resolution than the normalized spatial axis. An *eps*-value that would allow points from sequential months would have also allowed points from much wider spatial area. Thus, separate values suitable for individual axes were used. The cluster size of spatial-only DJ-Cluster proved promising, so same 0.001° *eps* value was used. To find seasonal effect, ±2 month *eps_time* was chosen. This allowed clusters to include points within few months and also join areas popular around the year. Because the *eps_time* further limit the amount of points in neighbourhoods, a smaller *min_pts* was used. In the preliminary tests, *min_pts* of 8 points provided suitable results. Once again, the analysis was first performed for a small, dense area in central Osaka as a test. After the preliminary test provided promising results, the analysis was performed for the whole dataset.

Table 1. Conducted analysis input values

Method	*Notes*
Spatial K-Means	k = 150, K-Means++ initialization, normalized space
Spatiotemporal K-Means	k = 151, K-Means++ initialization, normalized space
Spatial DJ-Cluster	*eps* = 0.001°, *min_pts* = 20
Spatiotemporal DJ-Cluster	*eps* = 0.001°, *eps_time* = 2 months, *min_pts* = 8

3.2 Visualization System

The intended audience for the results is a non-expert person researching for places to visit on a tourist trip. Accessing and browsing the data should therefore be as accessible and intuitive as possible. Thus, a system running on internet browser without needing installation of any extra plugins is a natural choice. Example screenshot of the system can be seen in Figure 7. For the time being, the proof-of-concept system is hosted at *http://joonamo.kapsi.fi/photoplaces/*.

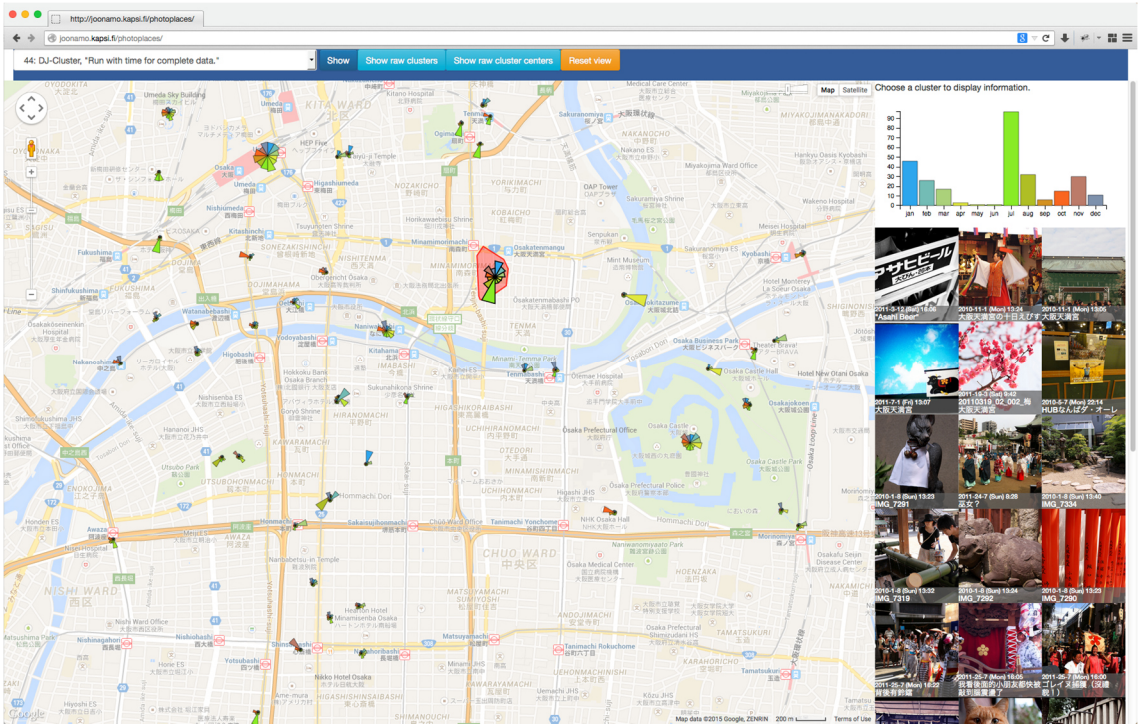


Figure 7. Browser-based visualization system. Cluster centres grow to 300% size when highlighted with cursor

The system was built in standard HTML5, CSS3 and JavaScript, so it runs platform-independently on pretty much any modern browser. The system was built using general-purpose utility library jQuery[1], Google Maps API v3 map library[2] and D3.js data visualization library[3]. These libraries are highly optimized, so zooming and scrolling the maps is fast and smooth. The system was designed to be intuitive and clean. Being a single-purpose system, it presents all available tools immediately. Main user interface consists of map view and info pane. User can choose a clustering run to display from a dropdown menu and it is added to map without a need of new page loads.

The glyph representing the clusters is designed to present not only the spatial location of the cluster centre, but also monthly point counts and the amount of points in the cluster relative to other clusters in the run. The relative point count is indicated by scaling the glyph. The glyph design is a modified radar plot. Different segments represent months a bit similar to clock. In the symbol, January is the first segment starting from top, proceeding clockwise. Size of different segments visualizes the amount of points in that month. The angle of the segments is always constant and only the area of segments varies. Every segment always has at least a small area to keep the cluster centre location clear. Colours of months gradually change and are chosen so they would be easy to be associated with different seasons. Cold blues were chosen for winter months, warm yel-

[1] http://jquery.com/

[2] https://developers.google.com/maps/web/

[3] http://d3js.org/

lows and greens for spring and summer months and reddish colours for the autumn. The same colours are used in both the cluster centre glyphs and the bar graph in the info pane for easier association. The glyph doesn't have any labels for the different months to prevent the map view from getting cluttered. Instead, the month names are shown on the bar graph.

The user interface is designed to be intuitive to use even without any training. When a cluster is highlighted with mouse, it scales to 300% size for easier inspection of the temporal distribution of points. When the user clicks a cluster centre, the map zooms to highlight that cluster's bounds and shows information of that cluster in the info pane. The info pane shows a bar graph presenting the point counts in different months. Info pane also shows random sample of example pictures from the cluster. User can click the thumbnails for bigger picture in the original photo sharing service.

3.3 Clustering Results

The clustering results for the test dataset produced by the used methods are presented and evaluated in this chapter. First part evaluates statistical measures of the clustering results and further subsections focus on the different methods individually.

Table 2. Cluster point count statistics. Clusters column presents total clusters produced by method, Max, Min, Average and Median present respective values of point counts in different clusters. Used% presents the percentage of points of the complete dataset in clusters

Method	*Clusters*	*Max*	*Min*	*Average*	*Median*	*Used%*
Spatial K-Means	150	4443	46	808.8200	516	100 %
Spatiotemporal K-Means	151	4471	52	803.4636	507	100 %
Spatial DJ-Cluster	100	6114	24	426.6100	144	35.2 %
Spatiotemporal DJ-Cluster	213	6971	8	238.7793	42	41.9 %

Both K-Means runs produced statistically very similar clusters. This can be explained by their practically same input *k*-value. K-Means by very definition always produces as many clusters as defined in the *k*-value and includes all of the points in data in result clusters. Therefore the average points in clusters will also always be *all points* / *k*, which can be seen in effect in the table. Consequently, the median value might be more descriptive measurement for K-Means clustering. Still, the median value is very similar for both runs.

DJ-Cluster runs, on the other hand, are statistically a lot more different from each other. First of all, with just spatial coordinates, analysis produced 100 clusters, while with the temporal coordinates included 213 clusters were found. Min and Median values indicate that the spatiotemporal analysis found a lot of small clusters, which is very appropriate considering the goal of the research. The higher amount of small clusters can be explained with lower *min_pts* value. This allows clusters to be created with fewer points. A somewhat strict *eps_time* variable was employed, so the amount of points in clusters didn’t get too high. Still, the spatiotemporal analysis used about 19% more points than just the spatial analysis.

Comparing the two different methods, few significant differences can be found. First of all, the median amount of points in DJ-Cluster runs is considerably lower than those of K-Means runs. This could be explained somewhat by K-Means not having the concept of noise and thus using more points in clusters. Interesting point to notice is that the cluster with most points isn't produced by either of K-Means methods, but by spatio-temporal DJ-Cluster. This cluster is in central Osaka, where the point density is very high. Both DJ-Cluster runs produced only one cluster there, unlike the K-Means variants, which produced multiple clusters. DJ-Cluster runs having lower minimum, average and median points in clusters, while still having most points in single cluster, suggests that DJ-Cluster produced a lot of smaller clusters and only few large clusters.

Table 3. Cluster area stats (km^2). *Only approximate values, calculated with fast and possibly inaccurate procedure*

Method	*Max*	*Min*	*Avg*	*Median*
Spatial K-Means	406.4833	0.3910	50.0530	21.7357
Spatiotemporal K-Means	485.0008	38.1705	157.0928	132.0497
Spatial DJ-Cluster	0.7804	0.0017	0.0457	0.0120
Spatiotemporal DJ-Cluster	1.2029	0.0000	0.0346	0.0061

Table 3 presents statistical information of cluster sizes created by different methods. Similarly to the cluster point count statistics, K-Means produced quite a lot larger clusters than DJ-Cluster variants. K-Means median and average clusters are many orders of magnitude larger than even the largest cluster produced by either DJ-Cluster variant. As an interesting note, spatiotemporal DJ-Cluster had the most points in any one cluster, but they are from a very dense area so the area of the resulting cluster is small compared to clusters generated by K-Means. K-Means runs produced a lot of large clusters that are not really useful considering the goal of the research. With higher *k*-value the K-Means runs would have created smaller clusters, but again having no concept of noise, the clusters might be even less meaningful.

Comparing the two different DJ-Cluster variants, a few interesting aspects can be found. First, the minimum area of DJ-Cluster with time axis is 0.0. This is not a rounding error, the cluster actually was a point-like where all the photos were reported to be taken at the same location. This might be caused by the inaccuracy of the phone GPS, especially if the photos were taken indoors. Due to lower *min_pts*, DJ-Cluster with time seems to have produced more small clusters than the spatial-only variant. Interestingly, it also produced the largest cluster of DJ-Cluster runs. As stated before, this cluster is located in very dense, all year round popular area, so low *min_pts* wasn't inhibited by strict *eps_time*. Everything considered, both DJ-Cluster runs created clusters with area very suitable for the research goal.

Table 4. Calinski-Harabasz Criterion for different runs in normalized spatial space

Method	*Spatial CH*
Spatial K-Means	199676.9443
Spatiotemporal K-Means	50451.9634
Spatial DJ-Cluster	7169928.1292
Spatiotemporal DJ-Cluster	3087742.3632

Calinski-Harabasz Criterion numerically analyses the clustering results and produces higher output value for better results. As CH is not well defined for cyclical values, only the fitness in spatial space was analysed. As stated earlier, CH prefers many small clusters, which differ from each other a lot. This can clearly be seen in the results in Table 4, as small-cluster producing DJ-Cluster runs have received orders of magnitude better results than K-Means runs.

It is worth a notice that both spatial runs have gotten better results than their spatiotemporal counterparts. For K-Means it is understandable, as both runs have almost the same number of clusters, but the spatiotemporal clusters are larger and exhibit a lot of overlapping. The overlapping can be seen in Figure 12. Spatiotemporal DJ-Cluster has statistically similar clusters as the spatial-only variant and doesn't have too much of overlapping. This might be caused by the spatiotemporal DJ-Cluster having more clusters around the mean centre of the data. CH seems to confirm what the statistical measures indicated; DJ-Cluster produced better results for this research goal.

Clusters created by different methods are explored and evaluated in the following chapters.

3.3.1 Spatial-Only K-Means Clustering Results

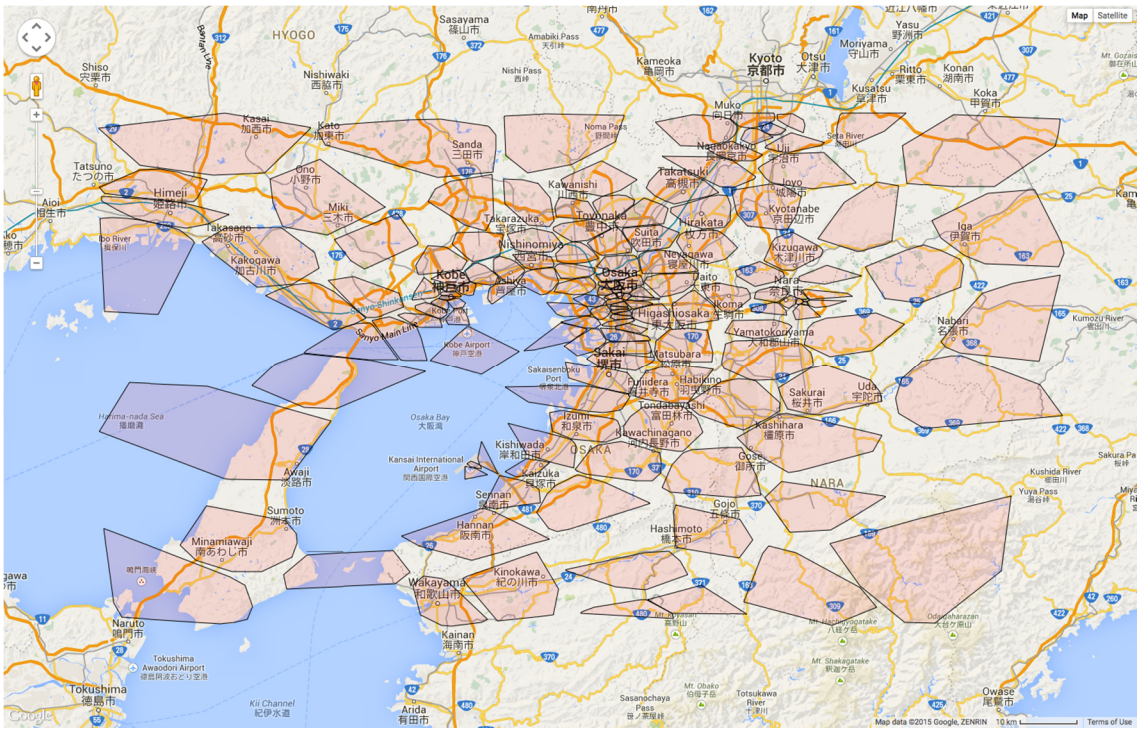


Figure 8. Cluster boundaries

Analysing Figure 8 clearly shows why K-Means isn't really suitable for the research goal; K-Means uses all the points in the data and therefore creates possibly large clusters. As stated earlier, having a concept of noise would probably help reducing the area of clusters. K-Means could be considered to be more about partitioning the data than actually finding clusters. The clusters are somewhat evenly spread across the research area. More clusters have been found in dense areas, such as Central Osaka, Nara and Kobe, but as displayed in Table 3, even those clusters have large area and don't really reveal relevant information.

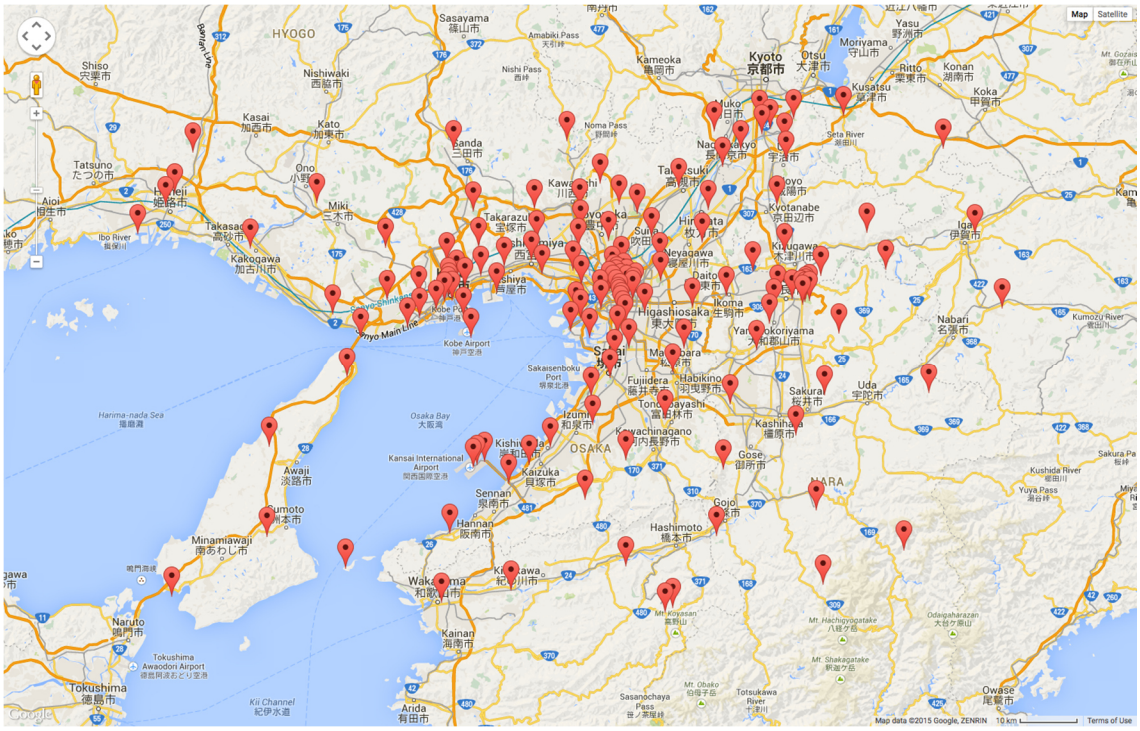


Figure 9. Cluster centre locations

Cluster centres displayed in Figure 9 and other maps displaying centres are positioned in the centre of mass of the corresponding cluster. Logically, it could be expected that the centre locations for at least the smaller clusters might point at a meaningful location. On closer inspection this appears to not be the case for K-Means clusters. Figure 10 a) and b) compares the closest cluster centres of two popular castles, "Symbol of the city of Osaka"[1], Osaka Castle and "Japan's most spectacular castle" [2], Himeji Castle. The cluster centre nearest to Osaka castle is actually very close to the actual castle, just tens of meters away. Compared to this, the closest centre to Himeji Castle is off about half a kilometre to north and seems to be pointing at a nursery instead of castle. According to japan-guide.com[3], the only other popular tourist spot in central Himeji is Kokoen Park, southwest from the castle, so the centre of mass isn't offset by that either.

[1] http://www.gojapango.com/travel/osaka_castle.htm

[2] http://www.japan-guide.com/e/e3501.html

[3] http://www.japan-guide.com/e/e3500.html

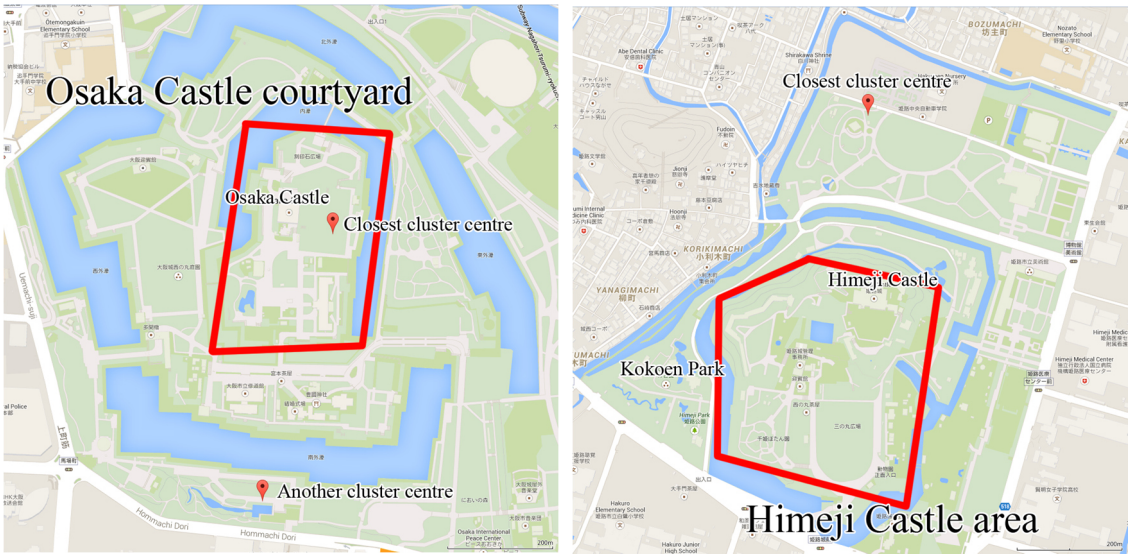


Figure 10. Closest clusters to a) Osaka Castle and b) Himeji Castle. Red areas are only for highlighting castle grounds and don't represent any particular cluster.

Other cluster centres confirm that the centre of mass doesn't point at a relevant location. Many clusters near bodies of water, such as sea or rivers, have their centre located in the water. In conclusion, clusters created by K-Means do not usually contain specific interesting places, but their centre of mass might be close to interesting place. Alas, the clusters are way too big to be meaningful and even the centre of mass doesn't reliably point at significant location.

3.3.2 Spatiotemporal K-Means Clustering Results

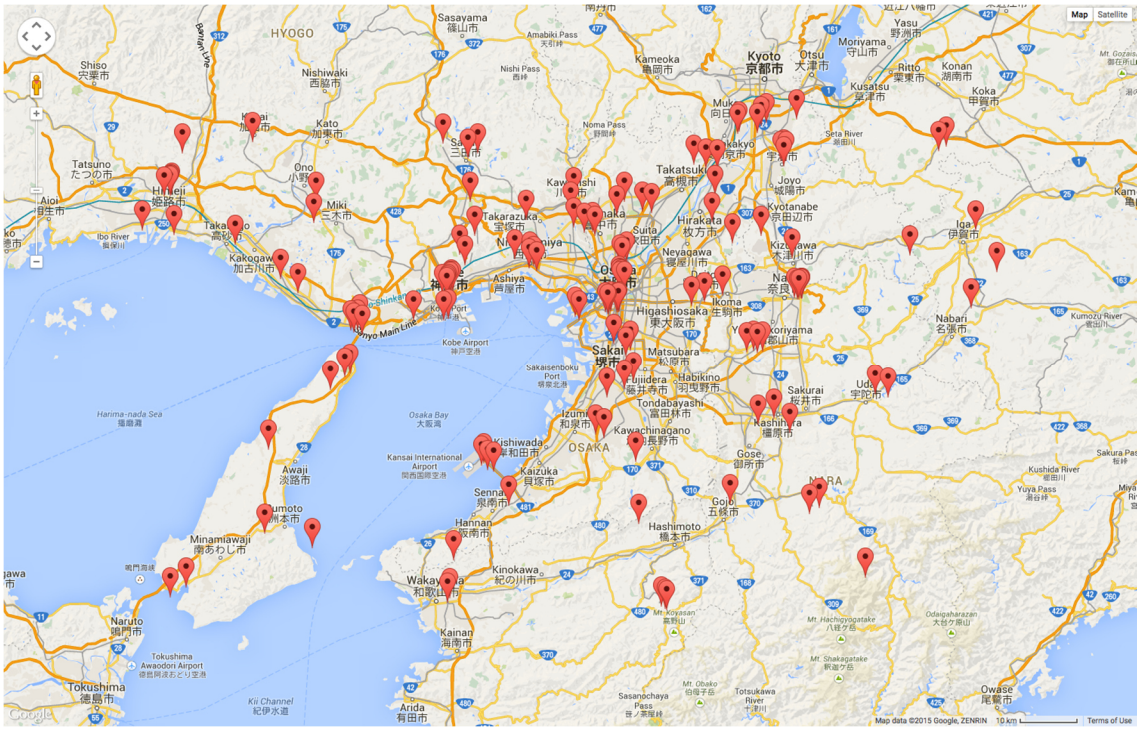


Figure 11. Spatiotemporal K-Means cluster centres

Even though both K-Means clustering runs were executed with very similar *k*-value, comparing the cluster centres in Figure 9 and Figure 11 show that their centres are spread quite differently across the 2d spatial coordinates of the research area. The spatial-only variant had the clusters rather evenly spread across the area with the denser areas slightly more densely populated by clusters. The spatiotemporal variant, on the other hand, has located the cluster centres more on the dense areas. Instead of many small clusters, this variant has found many spatially large and overlapping, topologically very similar clusters on dense areas. This can clearly be observed in Figure 12.

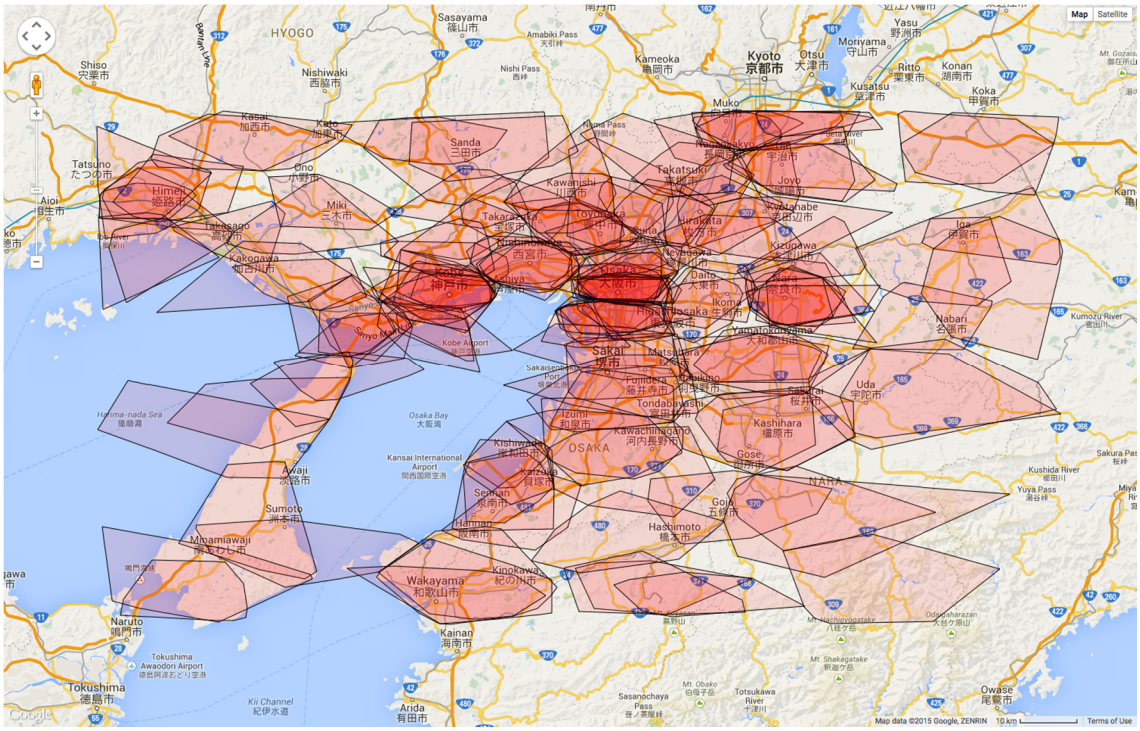


Figure 12. Clusters produced by spatiotemporal K-Means heavily overlap in spatial space

In this experiment, the time of a photo was only considered with one month resolution. In normalized space, one-month difference in time can be thought to be $1/12^{th}$ of the furthest spatial distance on either of the spatial axis. This resolution is quite a lot coarser than the resolution of spatial coordinates. Thus, in spatially dense areas, the temporal distance affects the 3 dimensional Euclidian distance the most. For example, in the densest area of the research area, many clusters contain points from only 1 or 2 different months. This outcome is highlighted in Figure 13. There the cluster centres and shapes are spatially very similar but differ temporally.

Due to similar *k*-value to spatial-only variant, but higher dimension count, the clusters are larger in spatial dimensions by average. Table 3 confirms this and even the smallest cluster is too large to represent any meaningful information. Similarly to spatial-only variant, cluster centres don't convey meaningful information either. In dense areas the cluster centres themselves are spatially clustered. Smaller clusters could have been produced with smaller *k*-value, but considering the problems with K-Means, even these smaller clusters probably wouldn't have presented significantly any better information. In conclusion, the results produced by spatiotemporal K-Means are not suitable to re-

search purpose. The results could be considered to be the worst ones out of the four analyses.

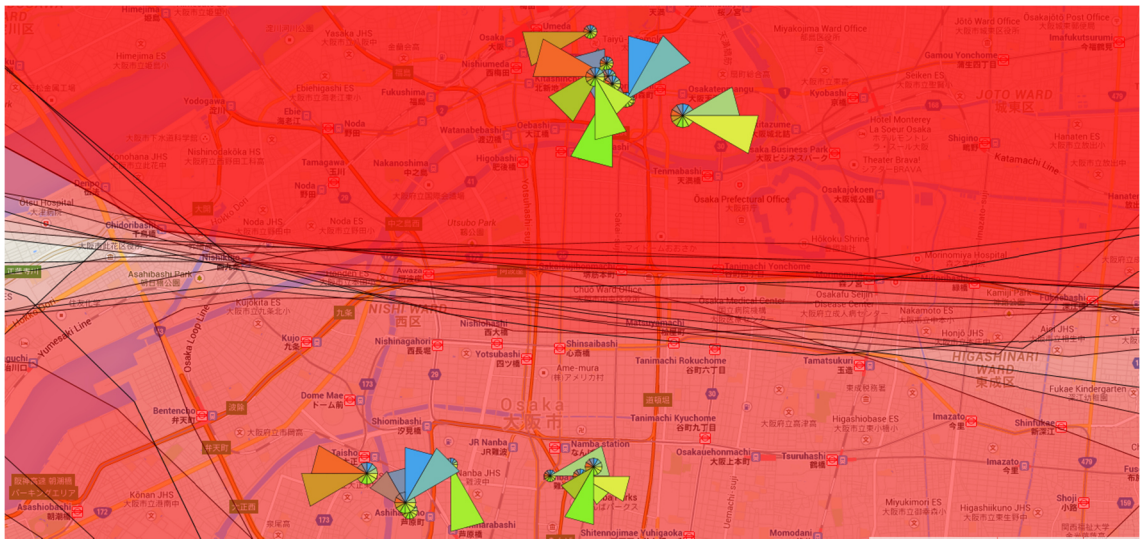


Figure 13. Cluster centres with temporal axis and cluster boundaries visualized in central Osaka

3.3.3 Spatial-Only DJ-Cluster Clustering Results

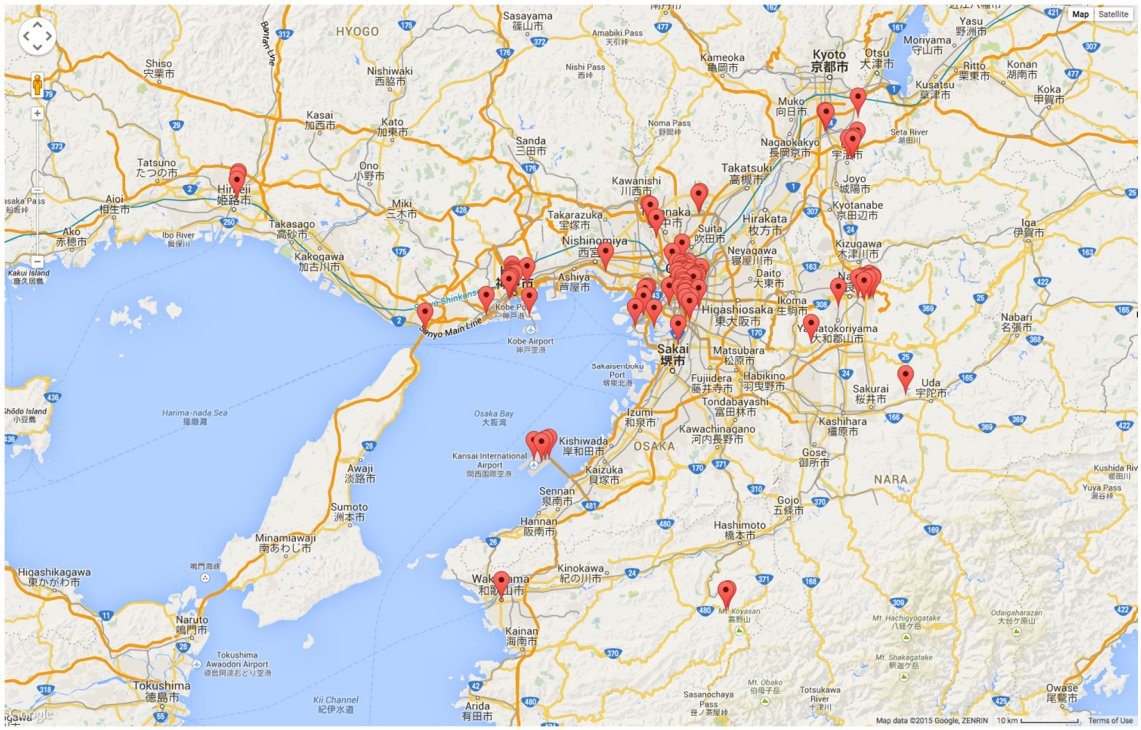


Figure 14. Cluster centres

Unlike K-Means results, DJ-Cluster doesn't need to include all input points in the resulting clusters, as points in sparse areas can be classified as noise. Figure 14 shows that the cluster centres are mostly located on the more dense areas. Large areas of map are left without single cluster found. Points too far from density neighbourhoods are classified as noise and don't show up in the results. Due to this, the clusters found are a lot

smaller than those of produced by K-Means, this can also be confirmed with Table 3. The clusters are so small that they are hard to see in overview map (Figure 15). The background map was dimmed in the figure to make the small clusters better visible. Small clusters are well in line with research goal, as they indicate hot spots that really could be interesting for person planning a tourist trip to the area.

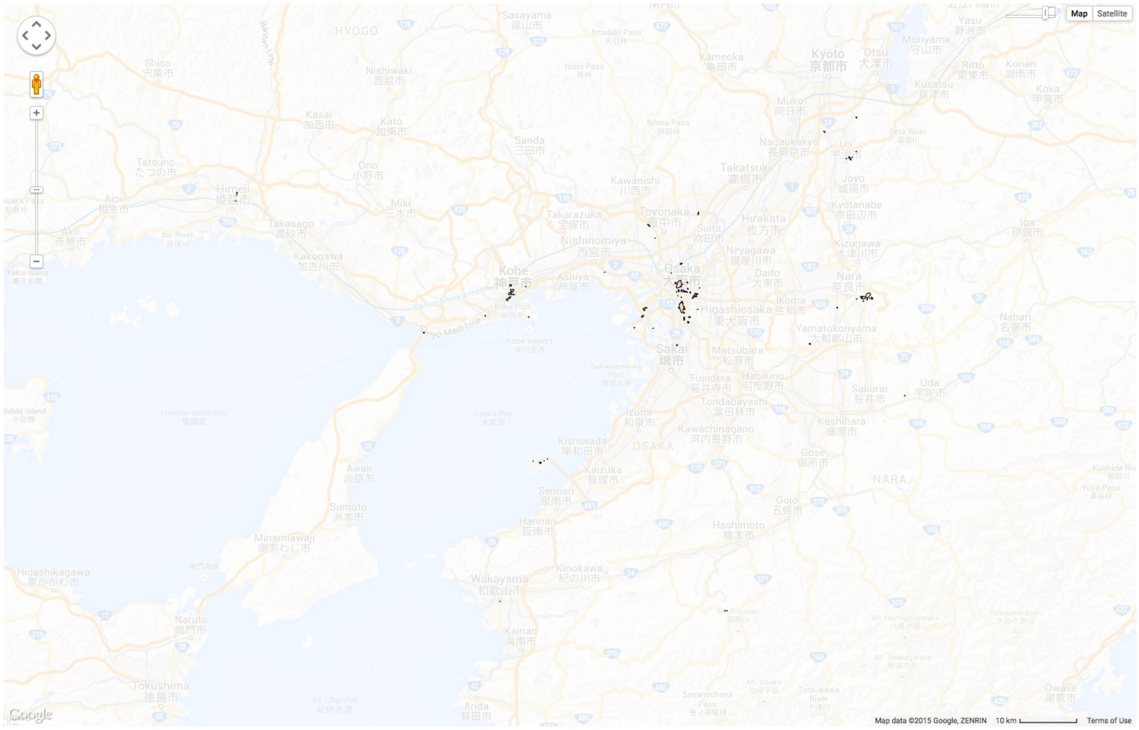


Figure 15. Overview of cluster boundaries

A more detailed view in Figure 16 gives a better representation of result clusters in central Osaka. Many smaller clusters have been found around the area. The two larger clusters are dense shopping area around Osaka and Umeda train stations (Northern cluster) and a popular tourist and shopping area of Shinsaibashi (Southern cluster). Other than those two, the clusters are relatively small and highlight different sights around the city. These sights include Osaka Aquarium, Universal Studios Theme Park, along with different parks and shopping areas.

Interestingly, multiple different clusters were found in Osaka castle area and Universal Studios Theme Park area. From both locations, the area around the main gate was highlighted. Around the Osaka castle few other significant areas can be found; the main castle courtyard, secondary entrance with nice view at the castle and different gardens. These could give meaningful insight for a person planning to visit the castle.

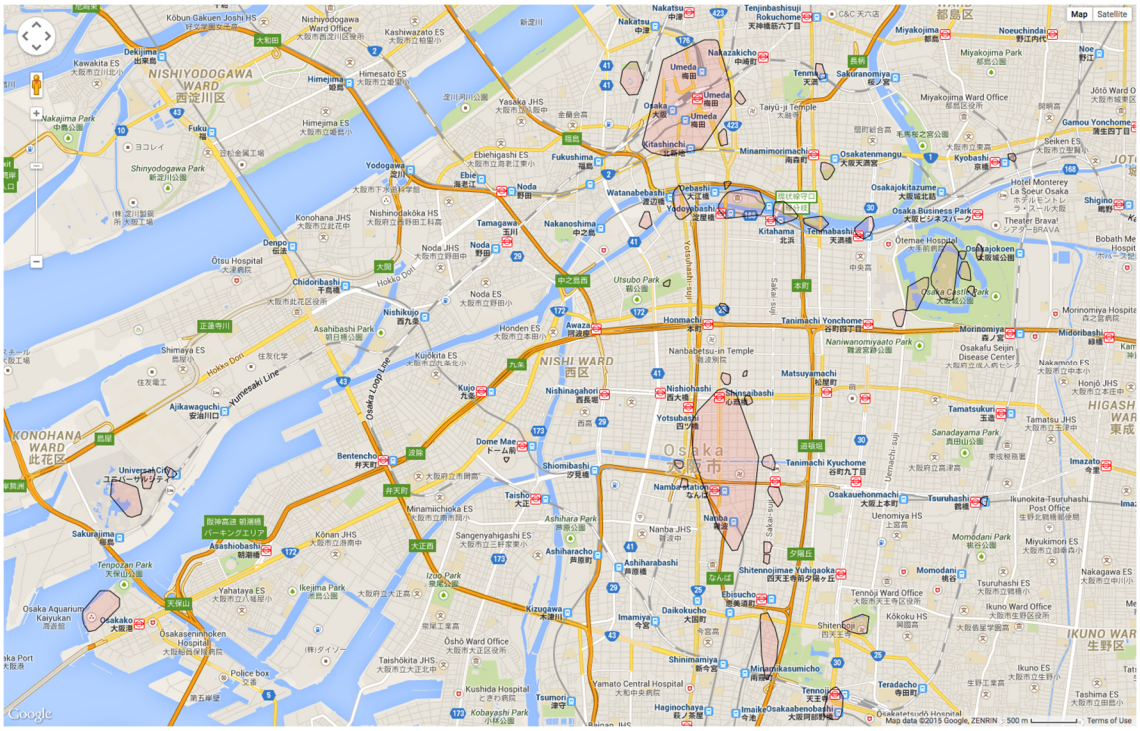


Figure 16. More detailed view of clusters in central Osaka

In conclusion, analysis results from spatial DJ-Cluster clustering are well suited for the research goal. The result clusters are small and dense enough to convey significant information about interesting places in the area. For the goal of this research, density-based methods seem like superior choice over distance-based methods. The result could probably be still improved by fine-tuning the input values.

3.3.4 Spatiotemporal DJ-Cluster Clustering Results

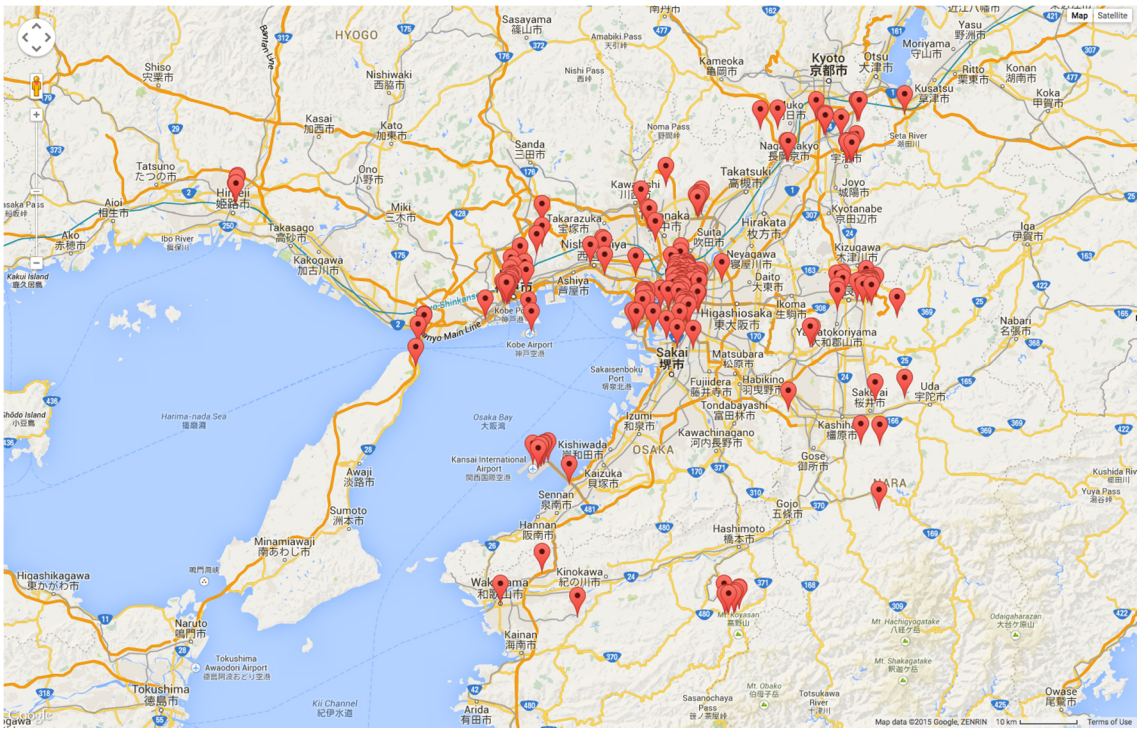


Figure 17. Cluster centres

Table 2 and Table 3 indicate that both variants of DJ-Cluster created statistically similar clusters. Visual inspection of Figure 17 and Figure 18 confirm that the results are similar, even though spatiotemporal variant took time axis into account and had lower *min_pts* variable. Due to lower *min_pts*, the spatiotemporal variant has found more clusters and smallest ones were even smaller than the clusters created by any other method. Similarly to spatial-only variant, clusters are not spread evenly across the research area, but are instead found mostly on dense areas. Even though the largest cluster found was a bit larger than the largest one in spatial-only variant, the median of cluster areas was smallest of all methods. Considering cluster area and point count medians, as well as visual inspection, this variant of DJ-Cluster could be considered to have produced the best results of all methods.

Unlike the results from spatiotemporal K-Means, the clusters generated by spatiotemporal DJ-Cluster only have minimal overlapping in spatial coordinates. Due to nature of the algorithm, DJ-Cluster tries to join neighbouring clusters together. Some overlapping can be observed In Figure 19, but the densest areas, namely Dotonbori area, are joined as just one cluster since the points can be reached in density-neighbourhood. Due to smaller *min_pts*, the Dotonbori area is a bit larger than in spatial-only variant. Still, it is considerably smaller in spatial space than the median clusters created by K-Means variants and can be considered to present meaningful information for a person searching for places to visit. One could for example deduce that “there is probably a lot to see in this bigger area”.

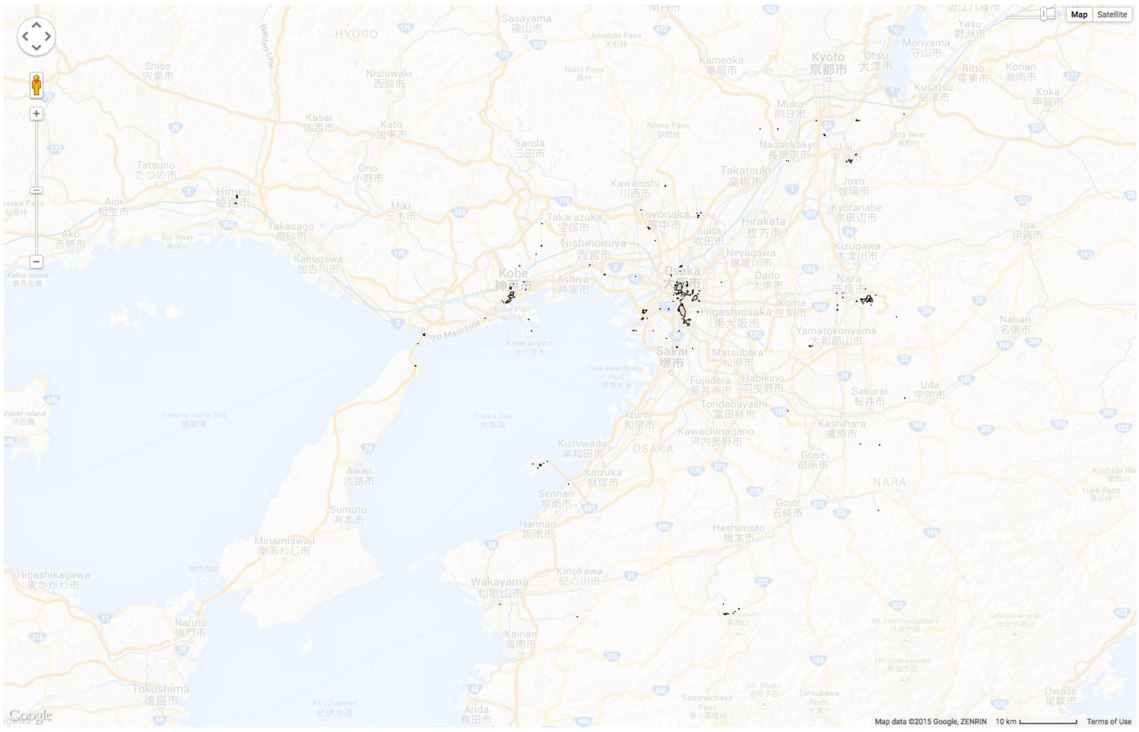


Figure 18. Cluster boundaries

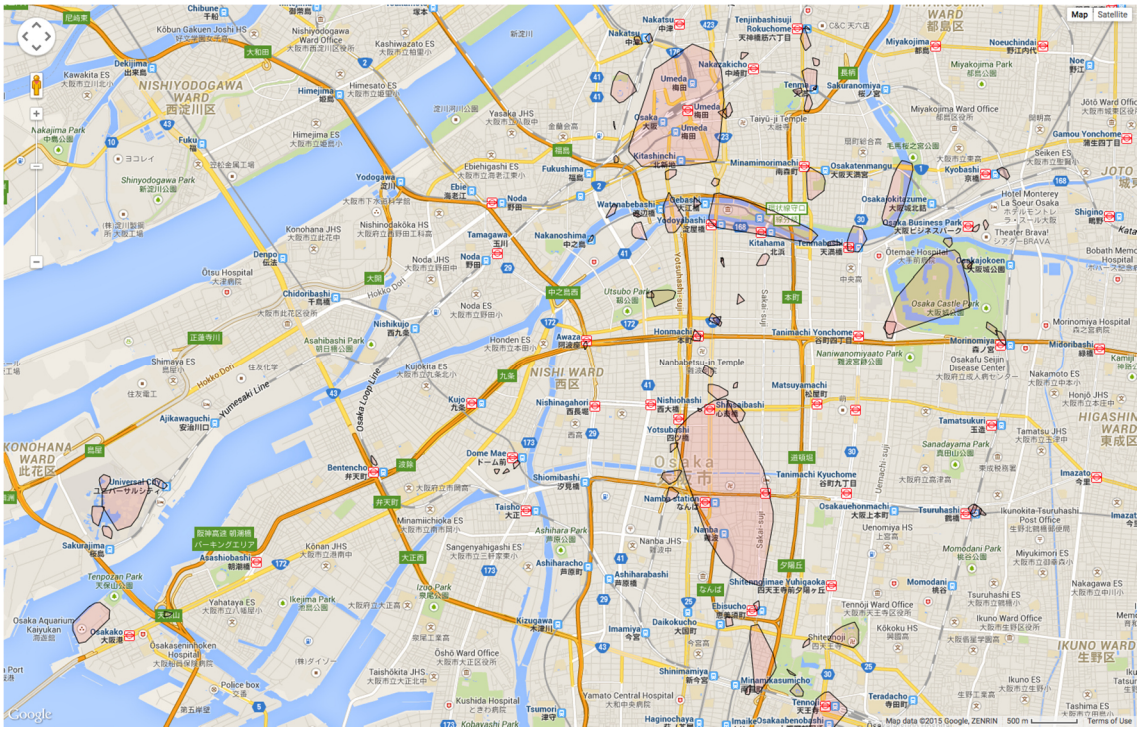


Figure 19. More detailed view of clusters in central Osaka

3.3.4.1 Seasonal Effect

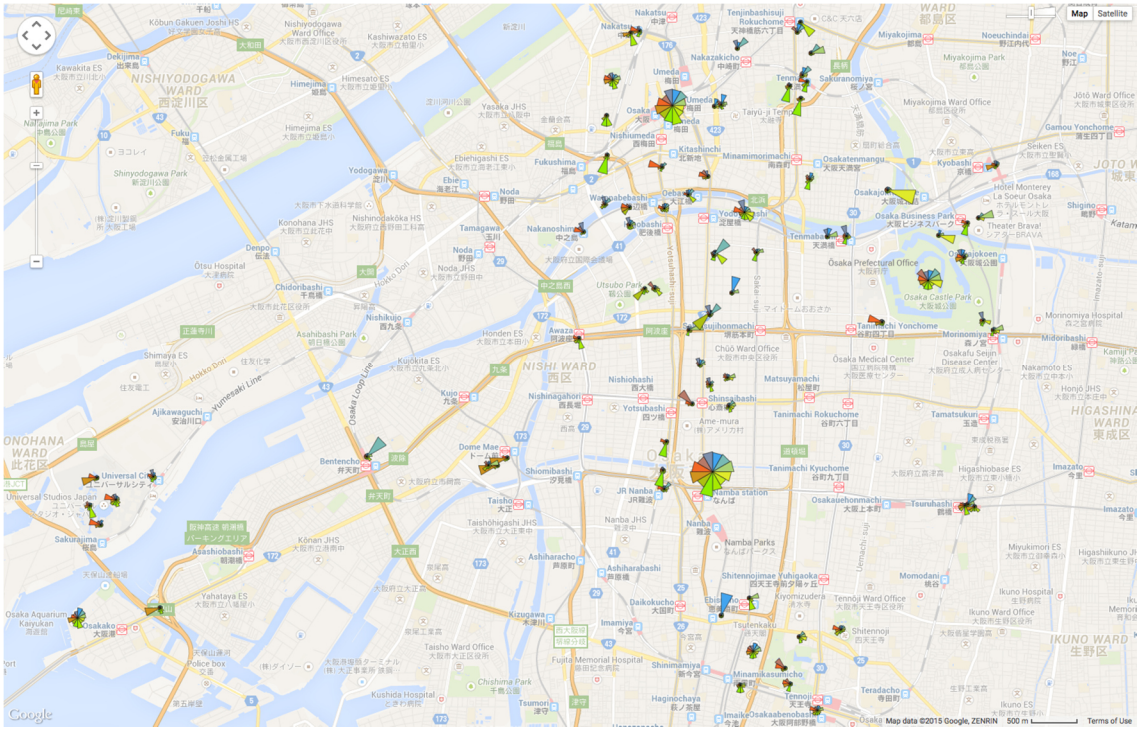


Figure 20. Temporal axis visualized for clusters in Central Osaka

This clustering method found many clusters that show signs of seasonal effect. At only ±2 months, the *eps_time* variable was quite strict. Many of the found clusters did only contain photos from a few different months. Figure 20 visualizes the spatiotemporal values of the clusters in central Osaka. Most of the clusters in this very dense area are seasonal. Similarly, clusters found in less dense areas are often seasonal. This chapter highlights some clusters that present interesting seasonal events and places around year.

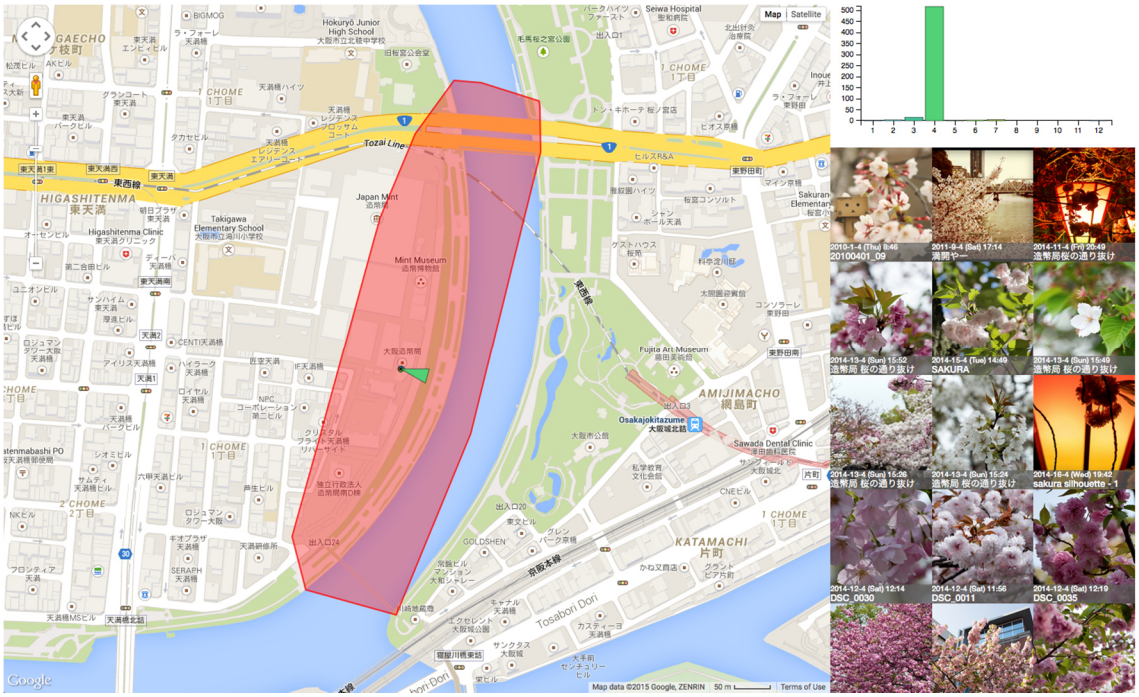


Figure 21. Hanami along a river

Figure 21 highlights one culturally important seasonal event found with this method. A long-standing tradition in Japan is to have a picnic under blossoming cherry and plum trees. The trees blossom only for a short period from the end of March to early May. This time is called "hanami" in Japanese, literally meaning flower viewing. Figure 21 highlights a popular spot in Osaka to enjoy hanami. This cluster's area is a strip along a river with only photos taken in March and May, the prime hanami season. Example photos also all display floral splendour of white and pink blossoms further indicating hanami.

Other similarly interesting seasonal clusters can be found while exploring data. Figure 22 highlights a popular New Year's festival in otherwise uninteresting residential area. Next, in Figure 23, a popular summer attraction, Kobe Oji Zoo, is featured. Curiously, the clustering has found two distinct clusters very close together. Photos in the eastern one focus mainly on maybe the biggest draw in the Zoo, the big panda, while the other has photos of many different animals. Finally, Figure 24 presents a popular area for the whole summer, garden of Osaka Museum of Natural History. Some users seemed very interested of photographing flowers and there are hundreds of photos of flowers in the sample data.

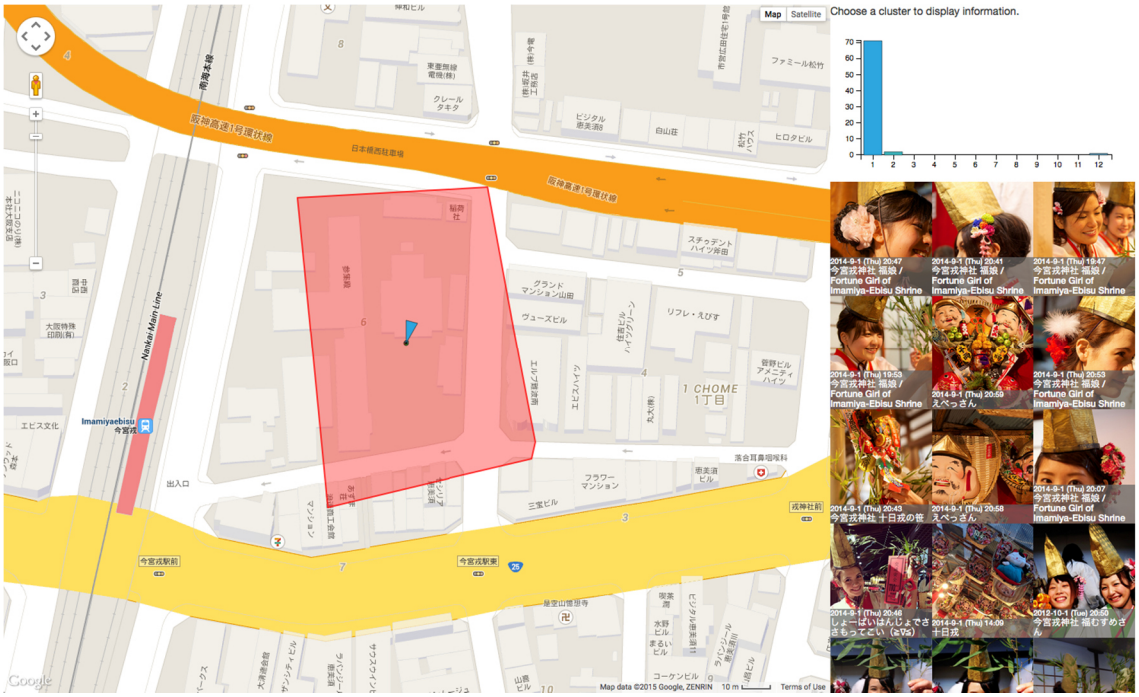


Figure 22. New Year's festival in residential area

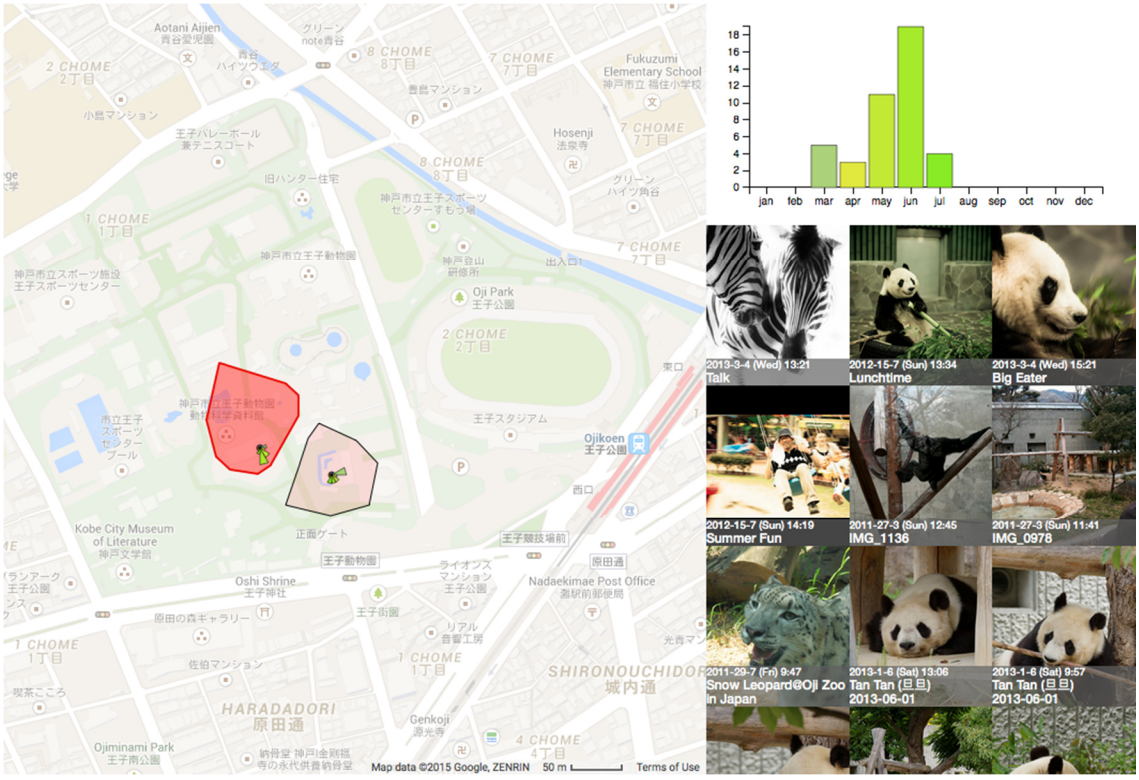


Figure 23. Kobe Oji Zoo

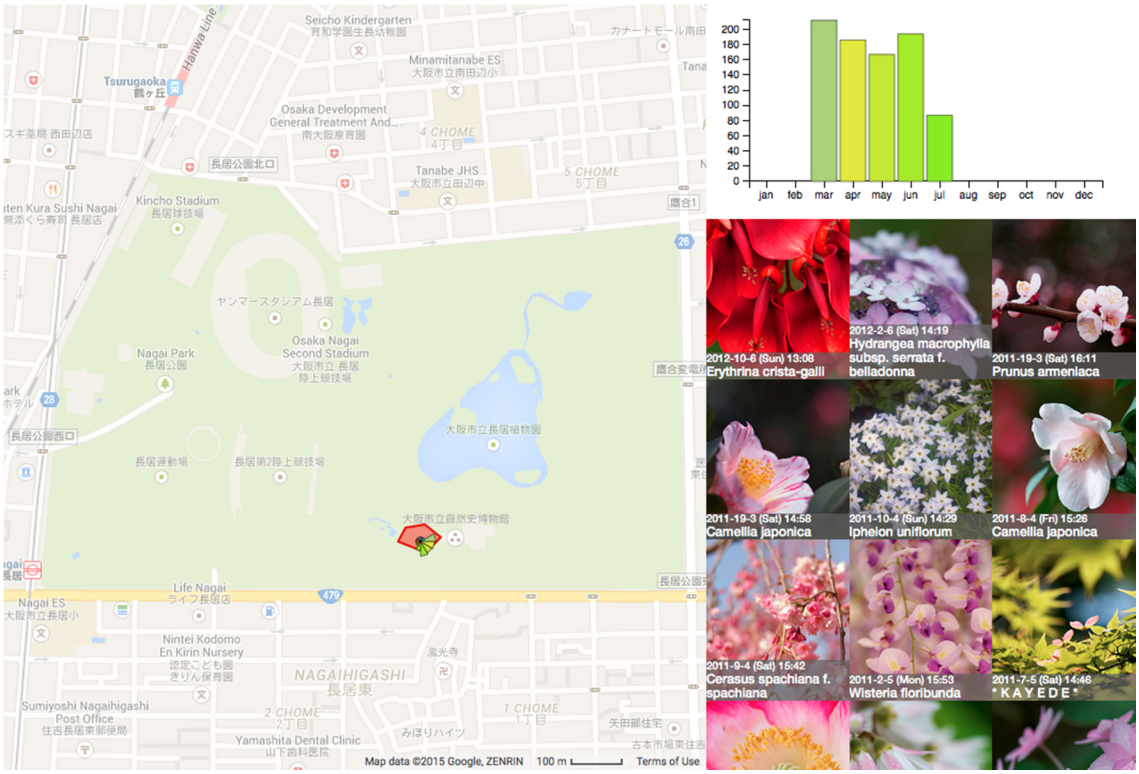


Figure 24. Floral Splendour of Osaka Museum of Natural History

In conclusion, the spatiotemporal DJ-Cluster found appropriately sized clusters similarly to the spatial-only variant. Many of the clusters exhibited seasonal effect. This chapter has highlighted some of them. DJ-Cluster with a strict-enough *eps_time* can be considered a very suitable algorithm for the purpose of the research.

4 Discussion

This chapter has two main purposes. First one is to evaluate the conducted work and the produced results against the research questions of this thesis. Second one is to discuss how the methods could be improved in the future research and potential applications for the theory presented in this thesis.

The first two research questions were about which methods can be used for finding interesting places and evaluating the effectiveness of these methods. This thesis presented several advanced pre-existent and derivative computational methods used as part of the research. From the two main clustering methods, DJ-Cluster prevailed with excellent results, while K-Means clustering can be considered unsuitable for the goal. The supporting methods, including methods for data normalization and processing cyclical time, fulfilled their function adequately.

In this research, time was processed as a cyclical value. Most of the reference material processes time as a continuous value instead. The used numerical computation framework, Python-based SciPy, didn't provide ready-made tools for circular values either, so required functions had to be built as part of the work. The presented methods for processing cyclical time worked well in practice. Used resolution of time variables was somewhat coarse, only one month. When normalized to z-score and compared with spatial z-scores, the resolution difference was considerably large. As a result of this, spatiotemporal K-Means produced few spatially similar and overlapping clusters with points from only one or two months. Finer resolution might have improved the results, but even the spatial only variant clearly displayed the method's fundamental incompatibility with the research goal, lacking the ability to filter out noise. Better temporal resolution might have also improved DJ-Cluster results, but the customized neighbourhood function with specific *eps_time* produced appropriate results.

In practise, the actual clusters created by K-Means were unsuitable for the research goal. The clusters were spatially too large to convey any meaningful information. As the text-book standard K-Means doesn't have any concept of noise, it created large clusters spatially covering just about the whole research area. Being this large, the clusters don't highlight any single interesting place. Even the cluster centres located in each cluster's centre of mass often didn't point to any meaningful place. The only input value for K-Means, the *k*-value, also proved problematic. K-Means always produces exactly *k* clusters, but in this case the amount of desired clusters to be found is unknown in advance. By increasing the *k*-value, more clusters with smaller area would have been found. Some of them might have pointed at interesting places, but there would be no way telling which ones exactly.

Contrary to K-Means, the other clustering method, DJ-Clustering, produced excellent results. By very definition, density based clustering methods only find clusters in dense areas. Large parts of the research area were left without a single cluster as the photos were too sparse in those areas. The found clusters were well aligned with the research goal; spatially small, dense and highlight singular places of interest. Both the spatial-only and the spatiotemporal variant produced equally good results without prominent spatial overlapping.

The other two research questions dealt with what kinds of places the analysis can find and if seasonally interesting places can be found. These questions are best answered in chapter 3.3.4.1, which highlights seasonally interesting clusters. Both DJ-Cluster methods found many different kinds of interesting places. The found places cover permanent establishments, such as castles, amusement parks and shopping areas. As a curious Japanese point of interest, many train stations were also highlighted. Spatiotemporal DJ-Cluster variant also found many seasonally interesting places. These also include permanent seasonally interesting establishments, such as flower gardens and a zoo, but also many events and festivals were found in the dataset. For example, religious events and traditional spring picnic areas were found by this method.

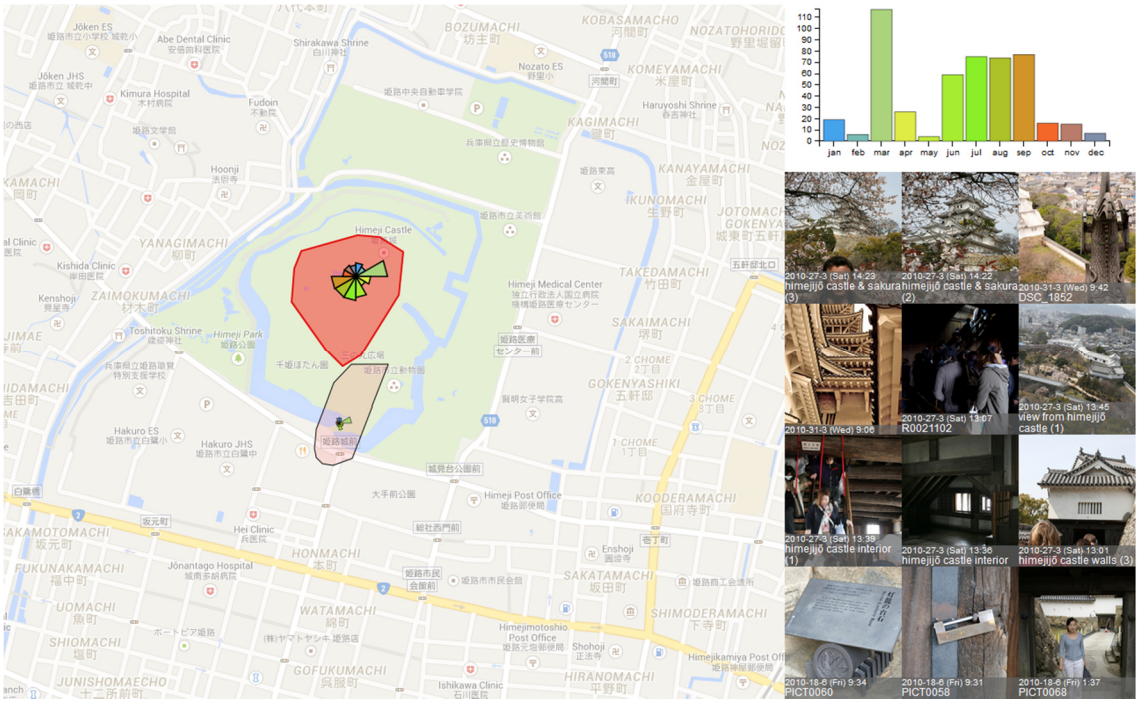


Figure 25. Clusters highlighting the gate and courtyard of "Japan's most spectacular castle" [1], Himeji Castle

The Python-based proof-of-concept implementation worked well enough for single study of pre-acquired data, but would require further development for a production quality system. First of all, the methods don't support continuous analysis of millions new photos uploaded each day. A further developed algorithm based on DJ-Cluster might be plausible choice for analysing continuously updated data. The algorithm is already based on the concept of joining density neighbourhoods together into larger clusters. In continuous implementation, a density neighbourhood could be calculated for every new photo and neighbourhoods could be joined when needed. Other possibility would be to periodically run the algorithm again, possibly processing only a subset of the data at once.

The algorithm input values are also somewhat problematic. In this work, the input values, *eps* and *min_pts*, were fine-tuned to work locally within the used dataset and re-

[1] http://www.japan-guide.com/e/e3501.html

search area. In global data, the densities of points will vary from place to place. Crandall et al. (2009) used non-parametric mean shift method in similar application. This approach wasn't adopted in this work as the mean shift results required further processing in their work. The algorithm also doesn't have concept of noise and implementation seems computationally intensive (Comaniciu and Meer, 2002). Some non-parametric density-based methods have been introduced, such as one by Azzalini and Torelli (2007), which is based on Delaunay triangulation concepts. Further research of different methods would be necessary for a truly global and continuous implementation.

The dataset of this work was limited to somewhat small area. Thus, the clustering results by themselves may not be very significant, but they clearly indicate that interesting places can be extracted from social media photos. Earlier research on similar subjects didn't consider time in the analysis. The acquired results indicate that seasonally interesting places and events can also be found by processing cyclical time as presented in this thesis. To further improve the results, the tags associated with photos could be used to determine the class of the depicted place. In their research, Van Canneyt et al. (2012) combined geotagged twitter messages, or "tweets", and geotagged photos from Flickr for deducting classes for places.

A potential party to employ methods presented in this thesis would be an image hosting service. Flickr, for example, already has a system in place to display pictures with high "interestingness" on a map[1] (Figure 1). Cursory analysis of the systems seems to indicate that their system doesn't employ any deeper analysis except just filtering recent and interesting photos on a given map area. Their system also doesn't suggest any spatiotemporally related photos for the user. The raw DJ-Cluster results already create clusters with photos of similar subjects, which could be used in this kind of exploration service. Some informal testing of the proof-of-concept visualization system was conducted with non-professional users interested in the research area. The testers found the results to be an interesting new way to find potentially interesting places and browsing photos of the area. For example Flickr's simple map might be more interesting to browse if there would be more spatiotemporally related photos available to browse along the highlighted ones.

[1] https://www.flickr.com/map

5 Conclusions

This thesis has explored and evaluated methods for finding interesting places for tourists from a large dataset of social media photos. The used photos had location information included in their metadata, which was be used as the basis for the analysis. The methods were evaluated on sample data acquired from Flickr. The photos were taken around Osaka, Japan.

Out of the two clustering methods evaluated, density-based DJ-Cluster presented by Zhou et al. (2007) produced excellent results. Unfortunately the other method, K-Means, produced unsuitable results. The result clusters identified by DJ-Cluster were small and densely populated with photos often pointing singular interesting places.

In this work, seasonally interesting clusters were also searched with spatiotemporal DJ-Cluster variant. This method also proved effective and found many spatially small clusters that are interesting only during specific time of the year. As density based clustering methods grow to include reachable points, this method was also able to also find places that are interesting around the year when appropriate. Spatiotemporal K-Means variant produced many spatially overlapping clusters instead.

In conclusion, DJ-Cluster can be used to find spatiotemporally interesting places from photo metadata. With further development to enable continuously updated and global data, for example a photo sharing service could benefit from these methods.

Bibliography

Arthur, D. and Vassilvitskii, S. (2007). K-Means++: The Advantages of Careful Seeding. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, pp.1027--1035.

Azzalini, A. and Torelli, N. (2007). Clustering via nonparametric density estimation. *Statistics and Computing*, 17(1), pp.71-80.

Berkhin, P. (2002). Survey Of Clustering Data Mining Techniques.

Comaniciu, D. and Meer, P. (2002). Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5), pp.603-619.

Crandall, D., Backstrom, L., Huttenlocher, D. and Kleinberg, J. (2009). Mapping the World's Photos. In: *Proceedings of the 18th International Conference on World Wide Web*. ACM, pp.761--770.

Flickr API, (2014). *Flickr Services*. [online] Flickr.com. Available at: https://www.flickr.com/services/api/ [Accessed 23 Oct. 2014].

Flickr, (2014). *Flickr: Explore interesting content around Flickr*. [online] Flickr.com. Available at: https://www.flickr.com/explore/interesting/ [Accessed 10 Dec. 2014].

Gramkow, C. (2001). On averaging rotations. *Journal of Mathematical Imaging and Vision*, 15(1-2), pp.7--16.

Hamerly, G. and Elkan, C. (2002). Alternatives to the K-Means Algorithm That Find Better Clusterings. In: *CIKM '02 Proceedings of the eleventh international conference on Information and knowledge management*. ACM, pp.600--607.

Kaufman, L. and Rousseeuw, P. (2009). *Finding Groups in Data: An Introduction to Cluster Analysis*. p.75.

Kisilevich, S., Mansmann, F., Bak, P., Keim, D. and Tchaikin, A. (2010). Where Would

You Go on Your Next Vacation? A Framework for Visual Exploration of Attractive Places. *2010 Second International Conference on Advanced Geographic Information Systems, Applications, and Services*.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* Berkeley: University of California Press, pp.281-297.

Mahajan, M., Nimbhorkar, P. and Varadarajan, *K*. (2012). The planar -means problem is NP-hard. *Theoretical Computer Science*, 442, pp.13-21.

Maimon, O. and Rokach, L. (2010). *Data mining and knowledge discovery handbook*. New York: Springer, pp.269-298.

Maulik, U. and Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(12), pp.1650-1654.

Michel, F. (2015). *How many public photos are uploaded to Flickr every day, month, year? / Combien de photos publiques sont téléchargées sur Flickr chaque jour, mois, année ? | Flickr - Photo Sharing!*. [online] Flickr.com. Available at: https://www.flickr.com/photos/franckmichel/6855169886/ [Accessed 21 Mar. 2015].

Seppänen, R., Kervinen, M., Parkkila, I., Karkela, L. and Meriläinen, P. (2005). *MAOL-taulukot*. Helsingissä: Otava.

Tango, T. (1984). The detection of disease clustering in time. *Biometrics*, pp.15--26.

Van Canneyt, S., Schockaert, S., Van Laere, O. and Dhoedt, B. (2012). Detecting Places of Interest Using Social Media. In: *WI-IAT '12 Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01*. IEEE, pp.447--451.

Virrantaus, *K*. (2013). *Multivariate data and explorative spatial analysis*. Lecture notes in GIS Analysis and Modelling - course 2013

Weisstein, E. (2014). *Sign -- from Wolfram MathWorld*. [online] Mathworld.wolfram.com. Available at: http://mathworld.wolfram.com/Sign.html [Accessed 19 Nov. 2014].

Zhou, C., Frankowski, D., Ludford, P., Shekhar, S. and Terveen, L. (2007). Discovering Personally Meaningful Places: An Interactive Clustering Approach. *ACM Trans. Inf. Syst.*, 25(3).